Complex product development model

Technical foundation

Introduction by
Christer Sandahl

For who is this introduction package ?

- This power point package is intended as
  - Cpdm executive briefing
    - for business managers and decision makers
  - Cpdm process cornerstones
    - for quality and improvement specialists
  - Cpdm technical walkthrough with examples
    - for technical managers and developers

Cpdm executive briefing

for business managers and decision makers

What is Cpdm ?

- Cpdm
  - Means Complex Product Development Model
- Cpdm model explain how to master development of complex products
  - Covers seamlessly mechanics, electronics and programs
  - Covers all phases from idea to established product
  - Covers both methodology and practical examples
  - Covers processes scalable to any product size
- Cpdm is inescapable complex but constructed very down to earth
  - After completing this presentation,
    the concluding feeling should be “how else could it be”

Why another model about this

- To master complexity require paradigm shifts
- Some industries have already succeeded
  - Space, aircraft, pharmacy, skyscraper, …
- Other industries has still far to go (programs in particular)
  - Artificial complexity being supported
  - Intrinsic complexity being ignored
  - Impossible principles applied
  - Watertight connection between technologies
  - Product structure undergoes degeneration
  - Products highjacked by engineers

Why is Cpdm the superior cure?

- Most other development models are only small scale
  - No explanations how development scale up to modern complex products
- Most other development models address only single technologies
  - No explanations how mechanics, electronics and programs fits together
- Most other development models describe only fractions of development work
  - No explanation how requirements, architectures, integration and verification fits together
- Most other development models are only theoretical and academic showpieces
  - No explanation and examples how to apply them on the floor
- Most other development models are only impenetrable amount of details
  - No explanation of how the details forms the superstructures
- Cpdm has none of above shortages

How to apply Cpdm ?

- Make assessment of complexity
  - Are own products considered to be complex ?
  - If so, do developers master such complexity ?
    - Are developers frustrated, is the quality lower than expected, is time and cost unpredictable, and so forth ?
  - If not, are management willing to improve ?
- Trying out Cpdm
  - Stepwise introduction, details bottom-up and/or superstructure top-down
  - Let train developers in Cpdm, use competent teachers with the Cpdm book
  - Use process employees and/or Cpdm consultants
  - Begin with experienced and motivated engineers
- Let the product rule
  - Evaluate if developed results meets higher expectation ?
  - Evaluate if products better satisfy customers expectations ?

Cpdm inventor Christer Sandahl

- Grew up in the Swedish entrepreneurship oriented “Gnosjö” area.
  - Not far from where IKEA’s Ingvar Kamprad grew up.
- Inventor since childhood
  - As child, staying in his grandfather's carpentry. Always playing with technical toys
  - When 14 years, own darkroom and chemical laboratory, later electronics with vacuum tubes and transistors
- Education
  - Master of Science in Electrical Engineering (M.Sc.EE.) at Chalmers University of Technology
  - Certified teacher in math and electrical topics for Senior high school
- Professional experience
  - Own and other small entrepreneur companies
  - Software manager at Axis, Lund
  - Management consultant at Q-Labs, Lund
  - Technical manager and operational development Ericsson and Sony Ericsson Lund
  - Wine producer of the best Riesling in Hungary
  - Board member in large family transport company

Book about Cpdm

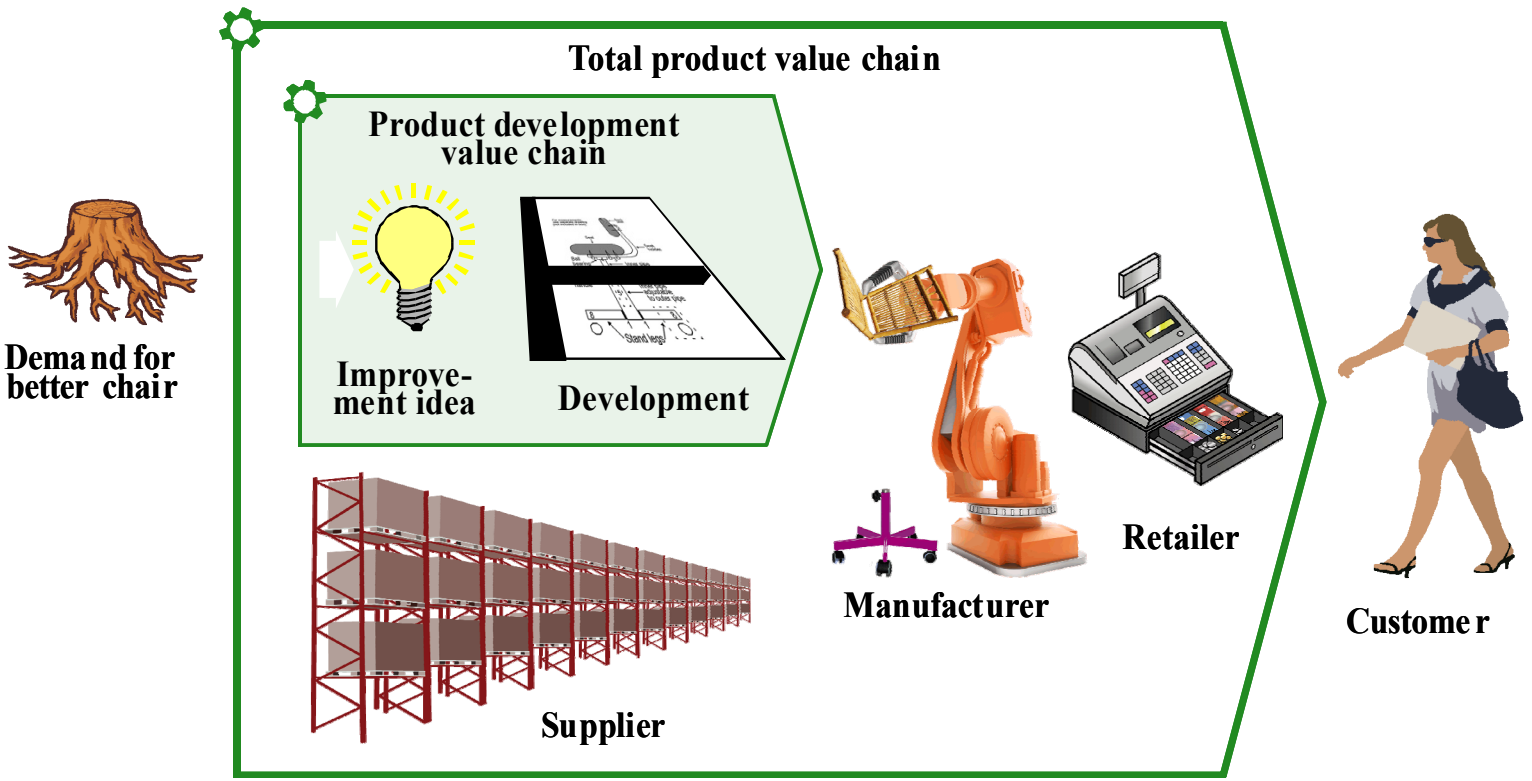
- First book about basic “technical foundation”
  1. Brief development from idea to established product of modest complexity
  2. Thorough development from requirements specification to prototype verification of complex products
     - 800 pages, 5 vast examples running along all phases, hundreds of pictures, figures and tables.
     - Extended support from Cpdm web site www.cpdm.com
- Planned books
  - Technical overhead (organization, management, quality, configuration management, etc)

Cpdm process cornerstones

for quality and improvement specialists

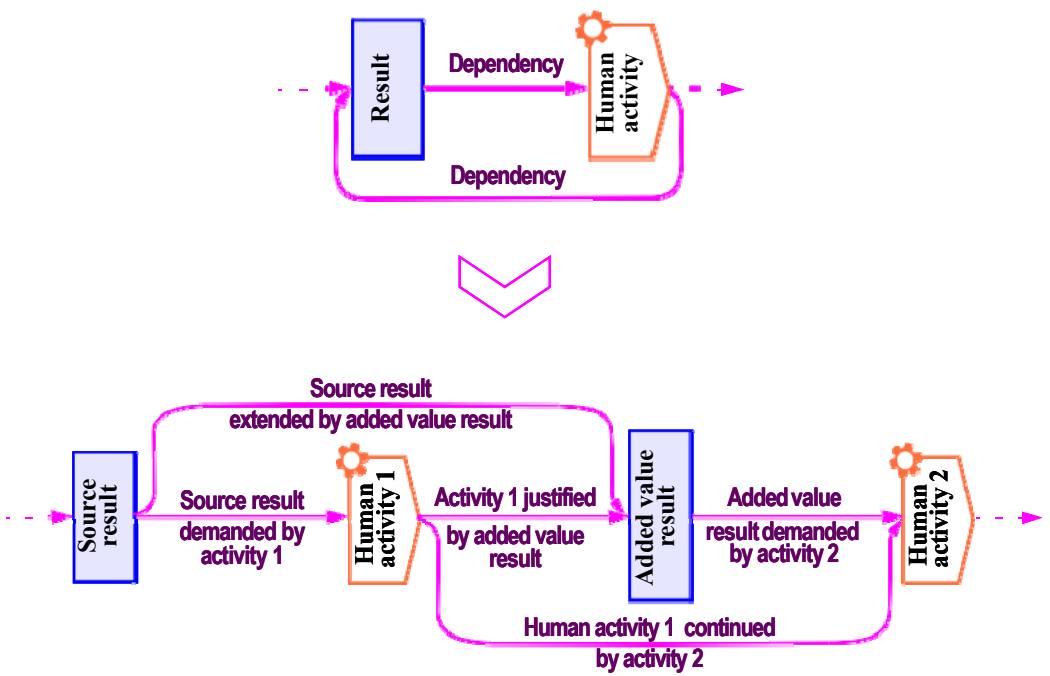
Total product value chain

- Development value chain is only one stage of the whole product value chain
- Cpdm covers the development stage

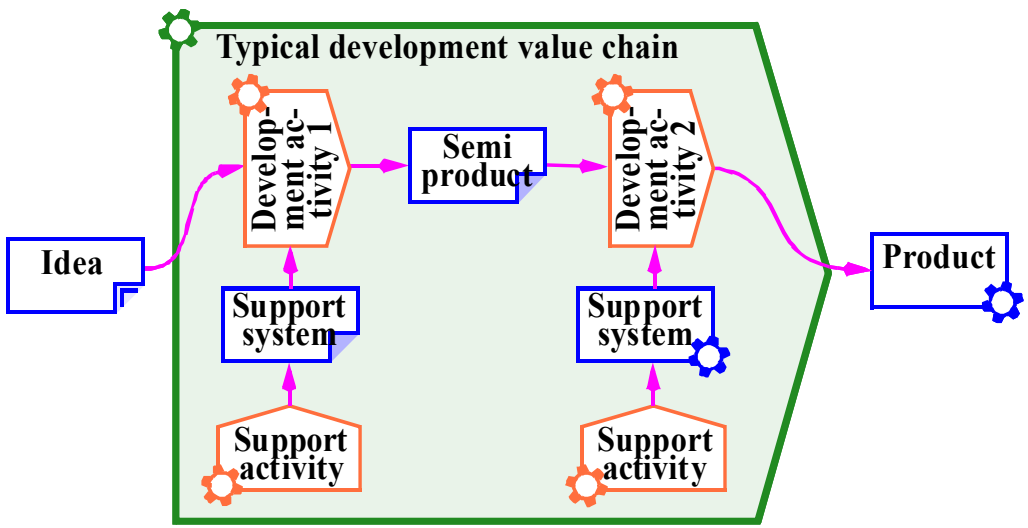


Value chain, activities, and results

- Activities, results and relations

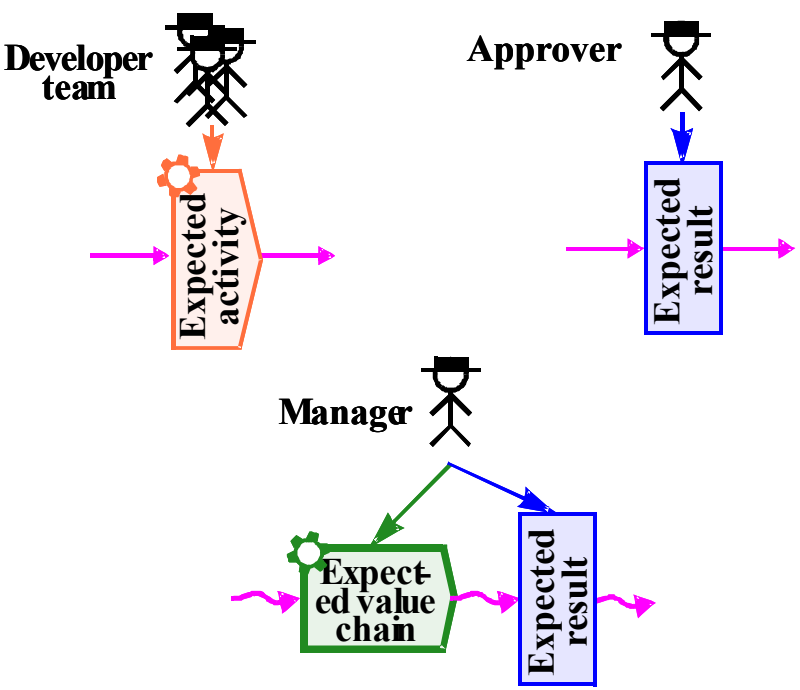


- Typical development value chain

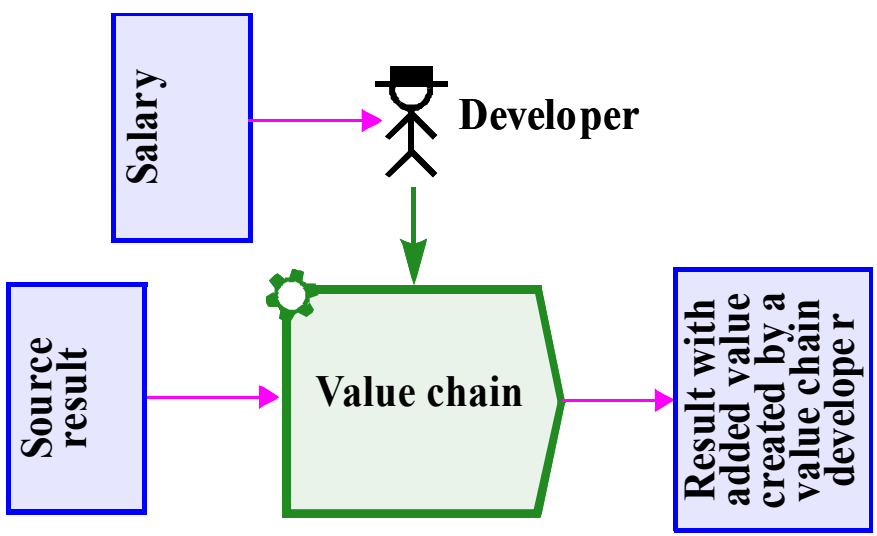


Value chain and roles

- Some generic roles in development

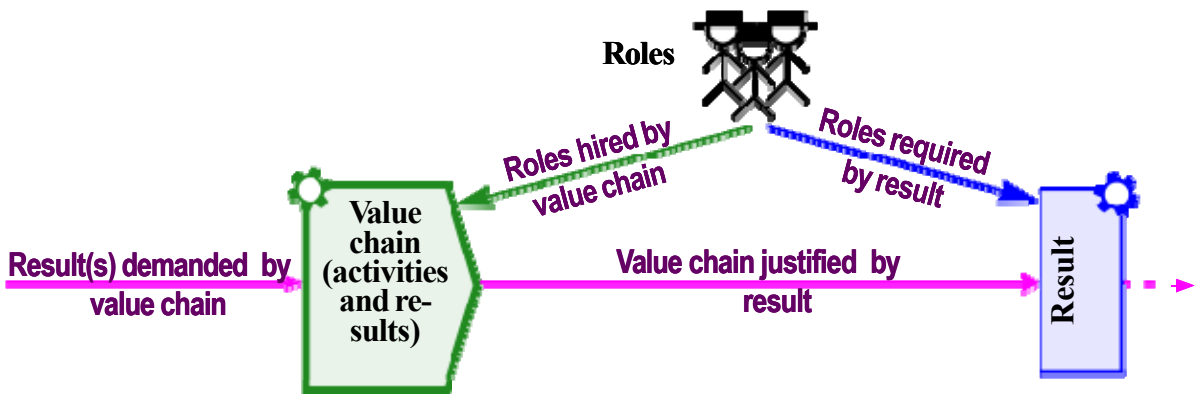


- In development the factual work adds value to a value chain result



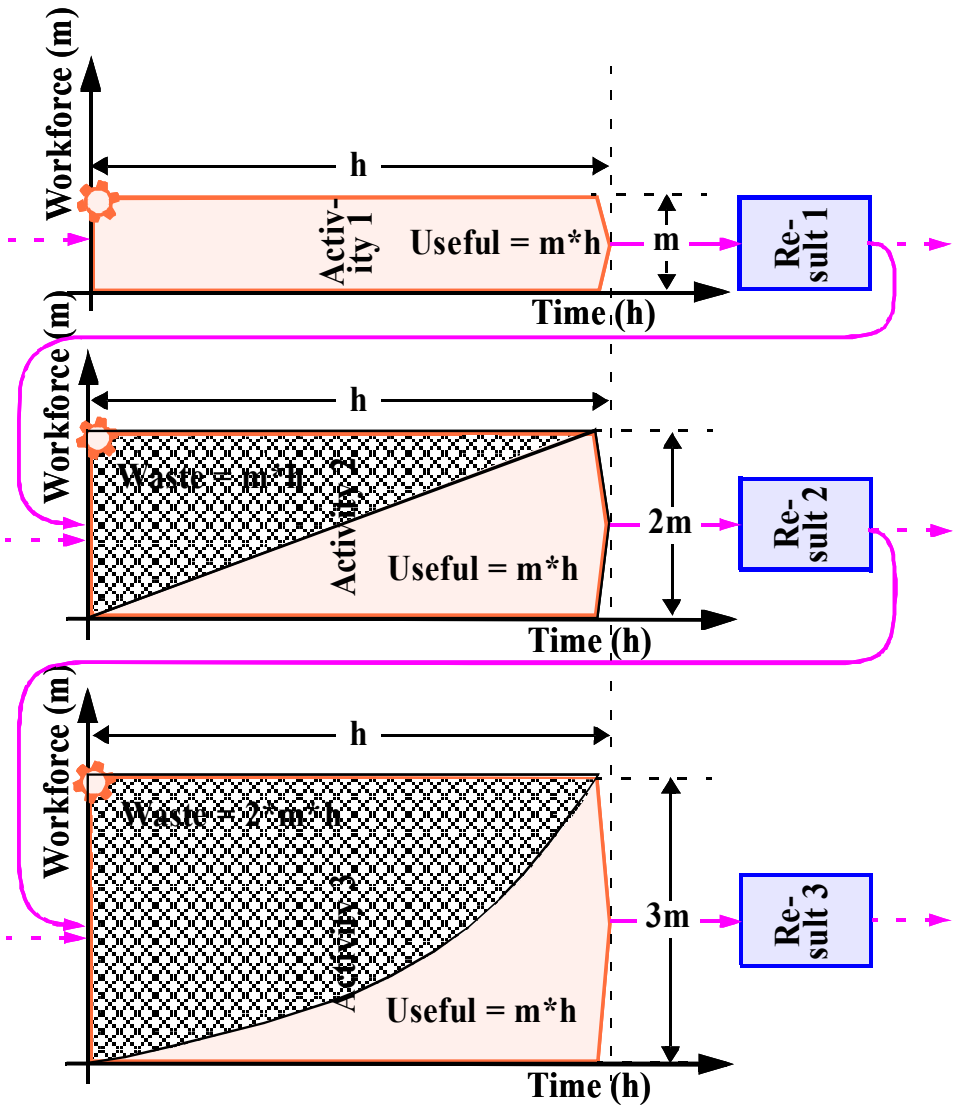
Value chain, results, and roles

- Relation between value chain, results and roles
- Should be read from arrow start to arrow head



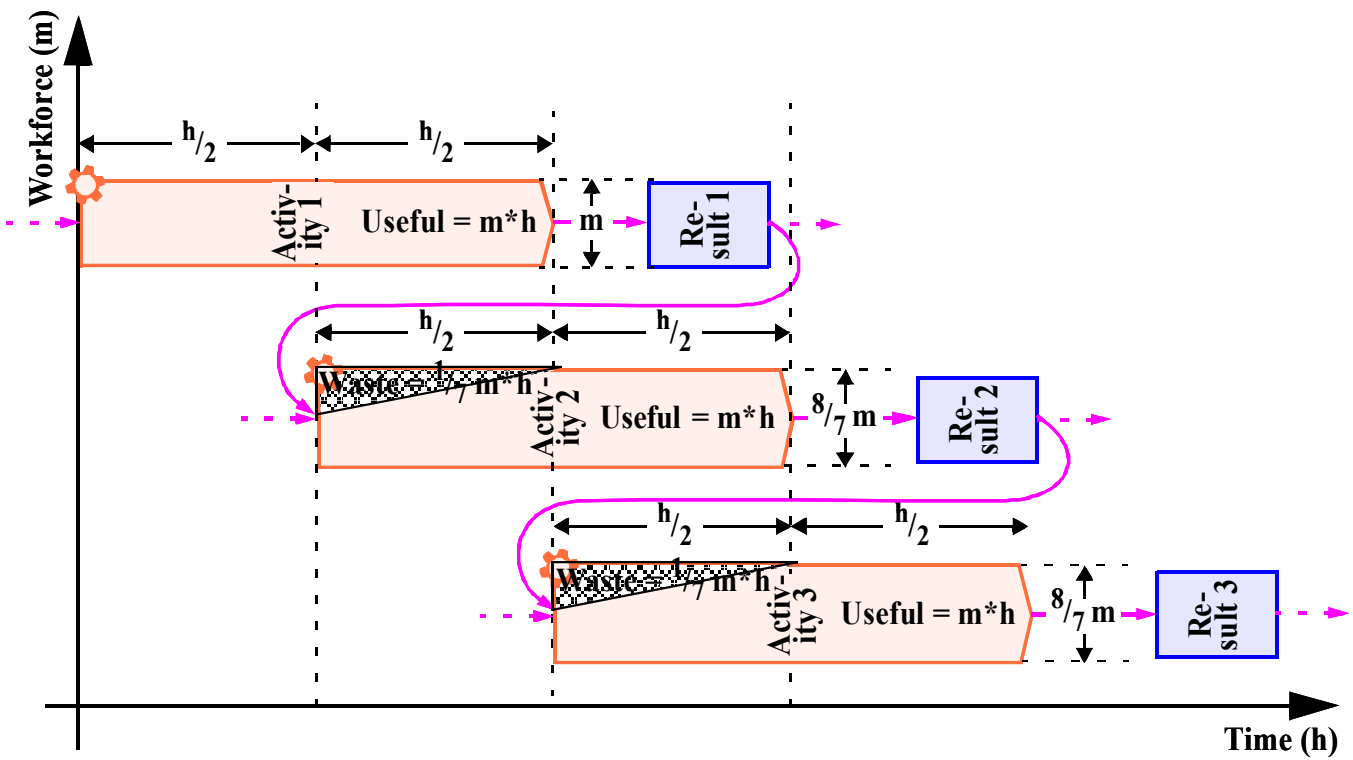
Execute three strict sequential activities in full concurrency

- Imagine that three activities have entire sequence dependencies and all of them are started simultanous
  - Activity 1 can start with available source results and no waste occur
  - Activity 2 can only guess at start, because result 1 is empty.
    - After half the time, result 1 is half-ready, and now activity 2 creates ½ waste
    - At full time, activity 2 has created m*h waste
  - Activity 3 can only guess at start
    - After half the time, result 2 is only ¼ ready, and now activity 3 creates ¾ waste
    - At full time, activity 3 has created 2*m*h waste
- A lot of waste is created



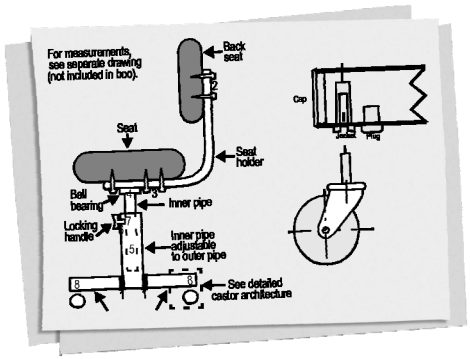
Execute three strict sequential activities when each activity starts when previous is half way

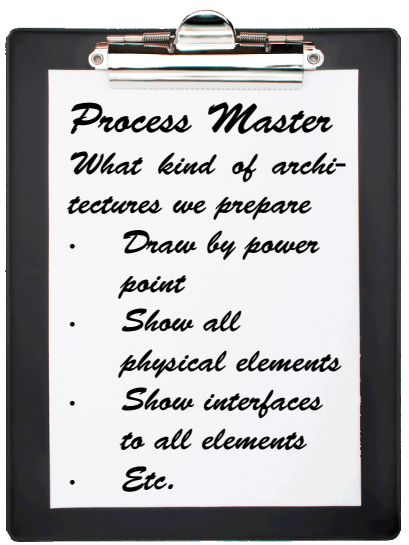
- By fully concurrence, a lot of waste is created
- Assume instead that 3 sequential activities start first when the previous is half ready
- Waste get = 1/7 * m * h (retained useful = m * h)
- Waste is limited and ratio not increasing with number of activities



Capture the value chain

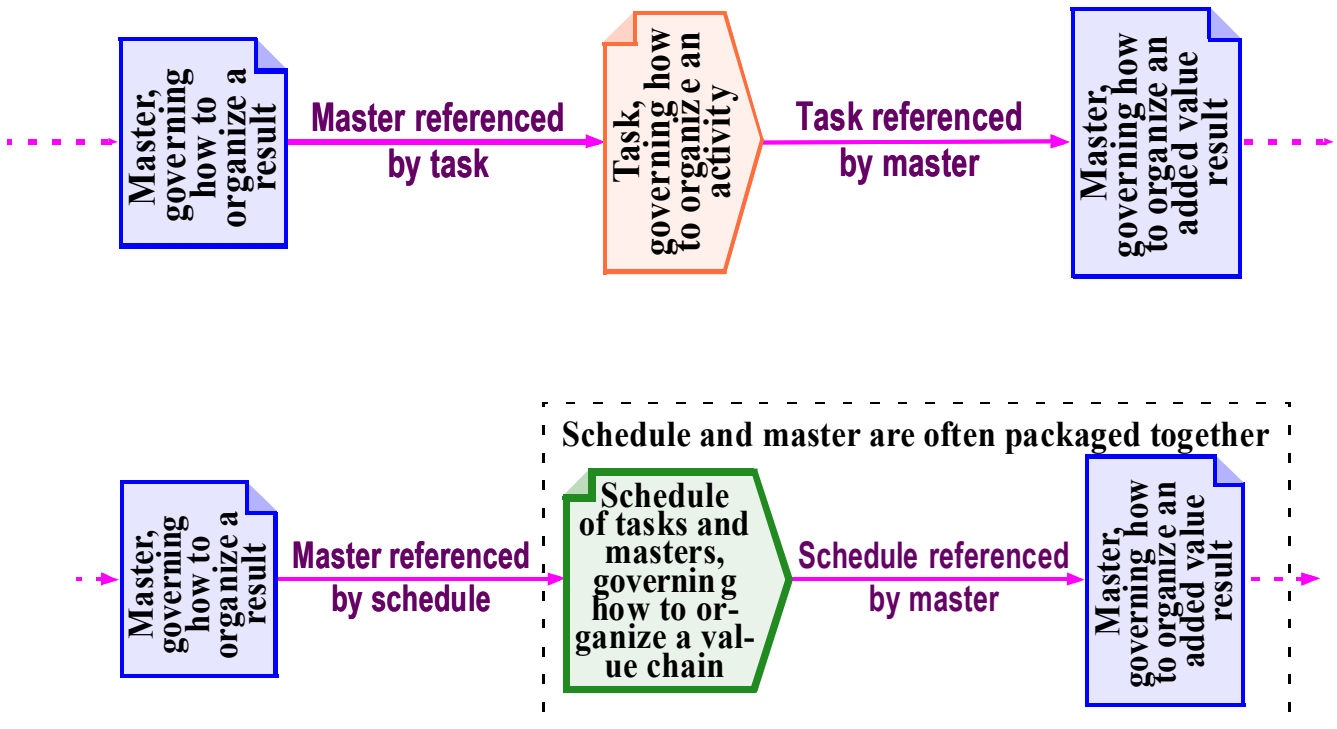
- Capture the value chain by process documentation
  - Activity described by task
  - Result described by master
- The value chain is much easier to handle when documented
  - Can be trained
  - Can be improved





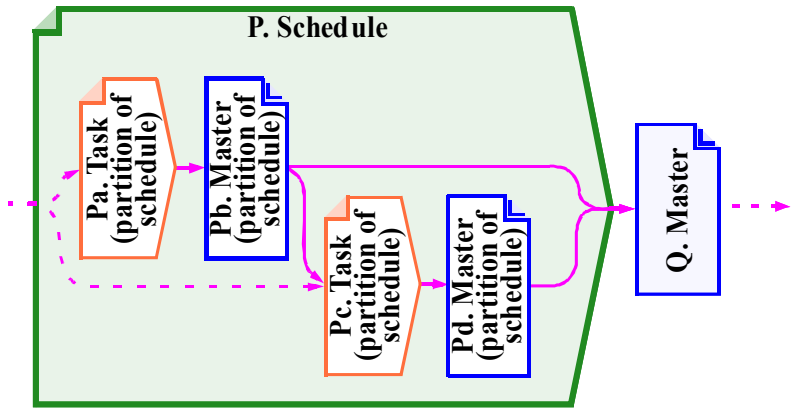
Process building blocks, schedules, tasks, and masters

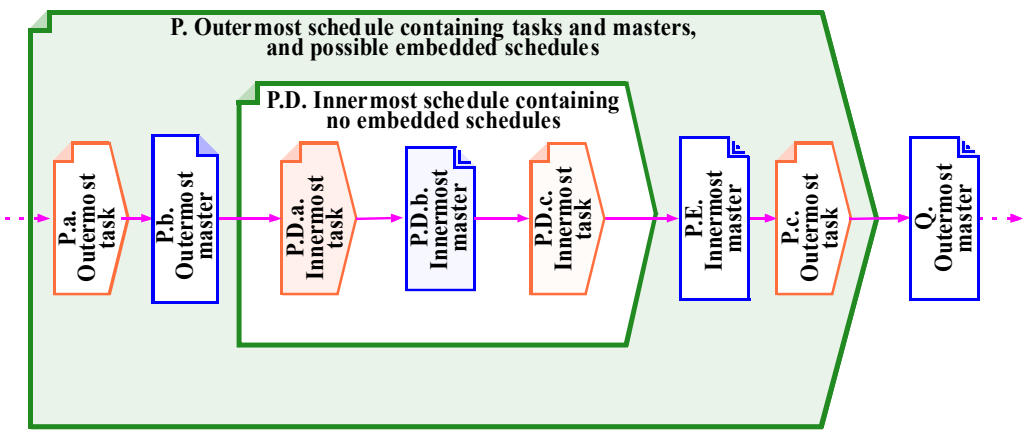
- Symbols for task, master and schedule
  - Task is a description governing how to organize an activity
  - Master is a description governing how to organize a result
  - Schedule is a package of masters and tasks, see next slide



Schedule aggregation and hierarchy

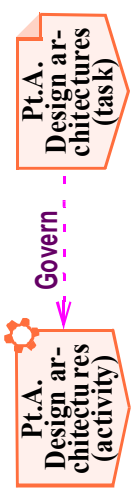
- Schedule aggregates tasks and masters
- Schedules scale up by aggregate embedded schedules
- Schedule identity characters reflect level in hierarchy (as in an architecture)

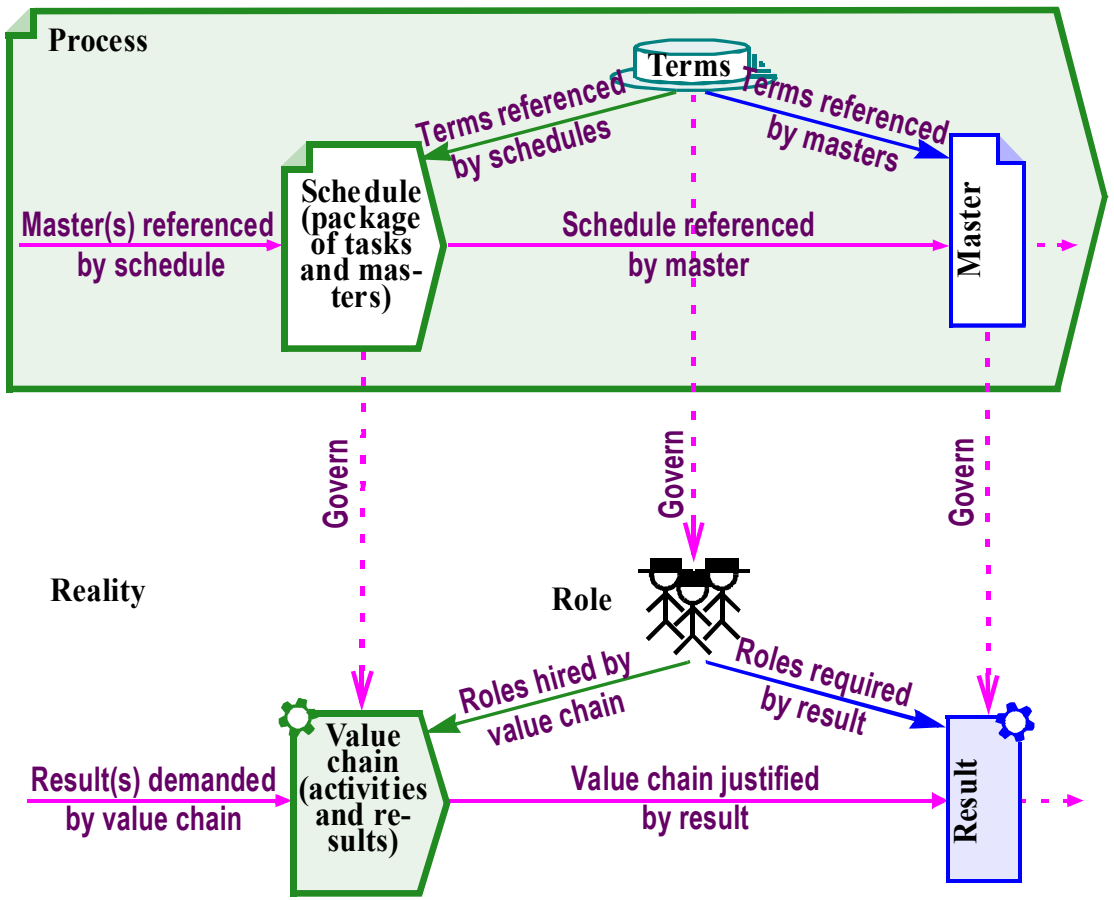




Process and reality relation

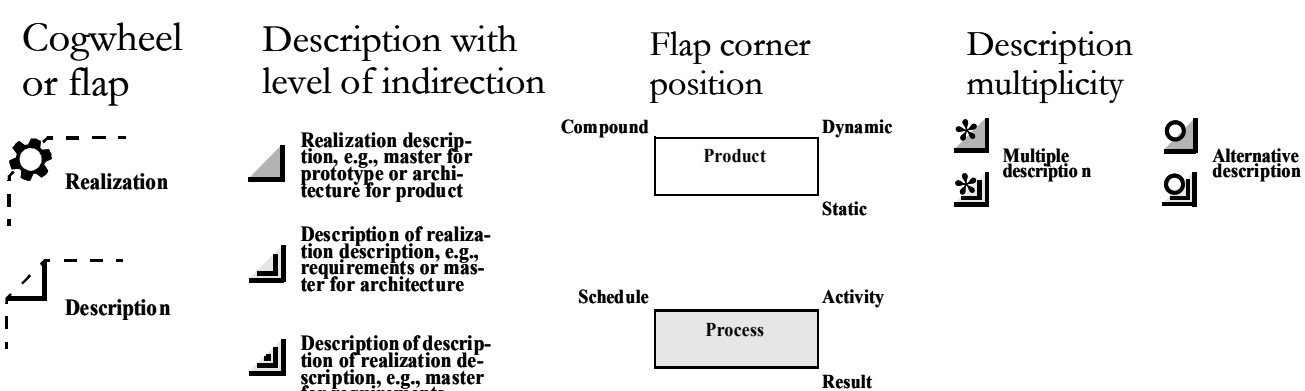
- Process governs the value chain
- Cpdm illustrates the difference between process and reality
- Same names in process and reality

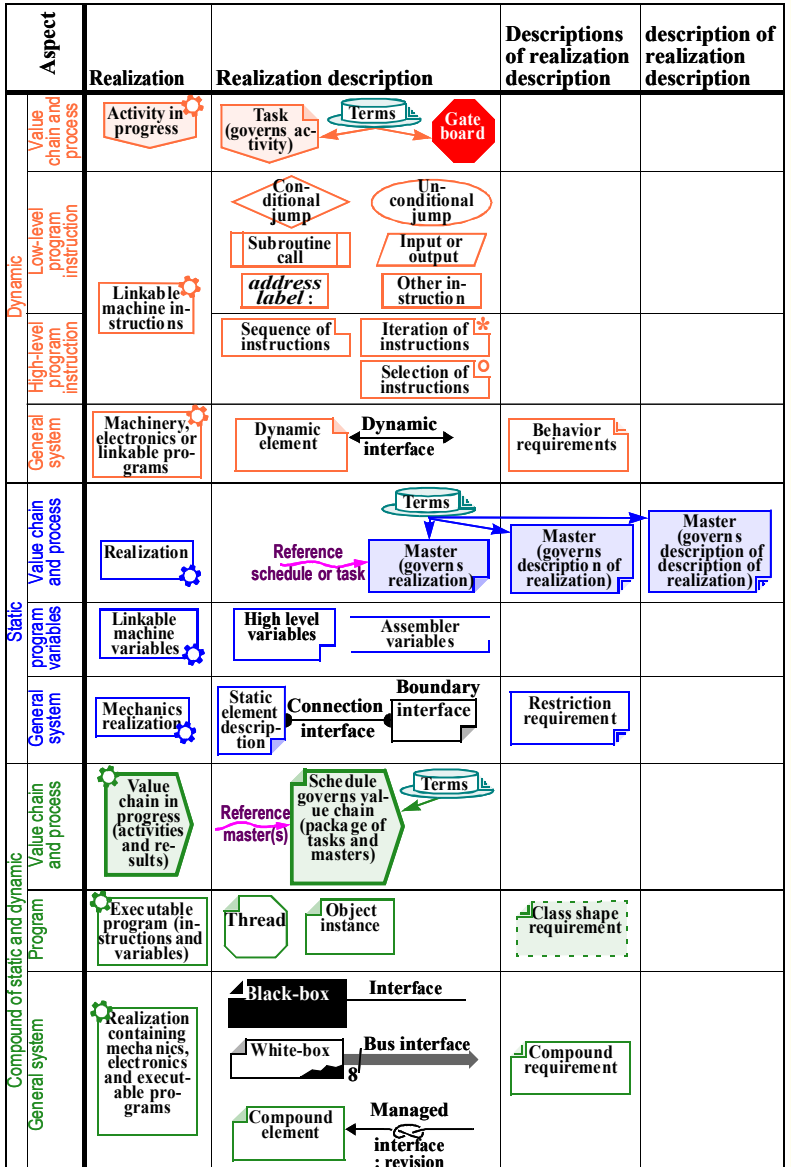




Cpdm graphical symbols

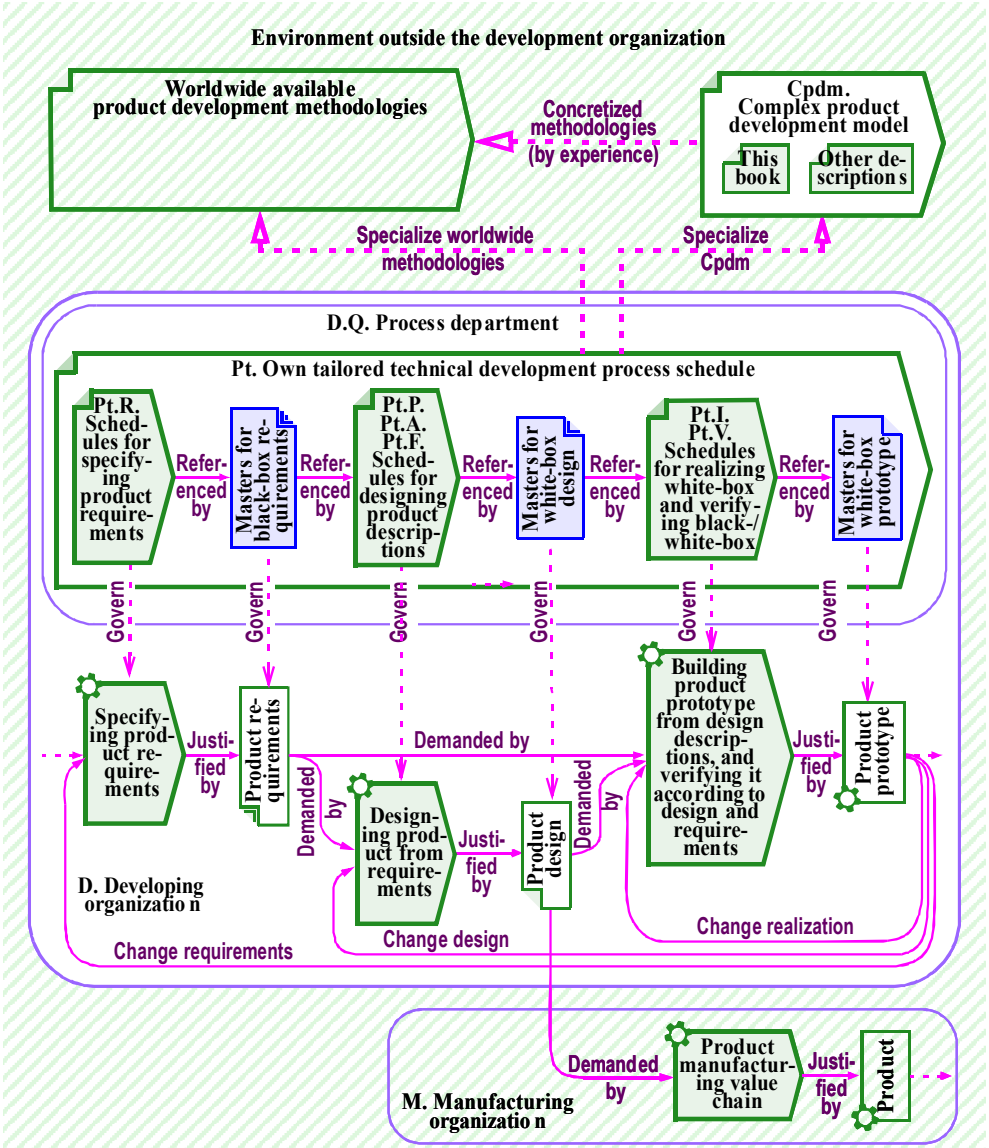
- Cogwheel or flap indicate reality or description
- Description may be of different level of indirection
- Flap position indicates static, dynamic or compound
- Multiplicity flap indicates iteration or selection





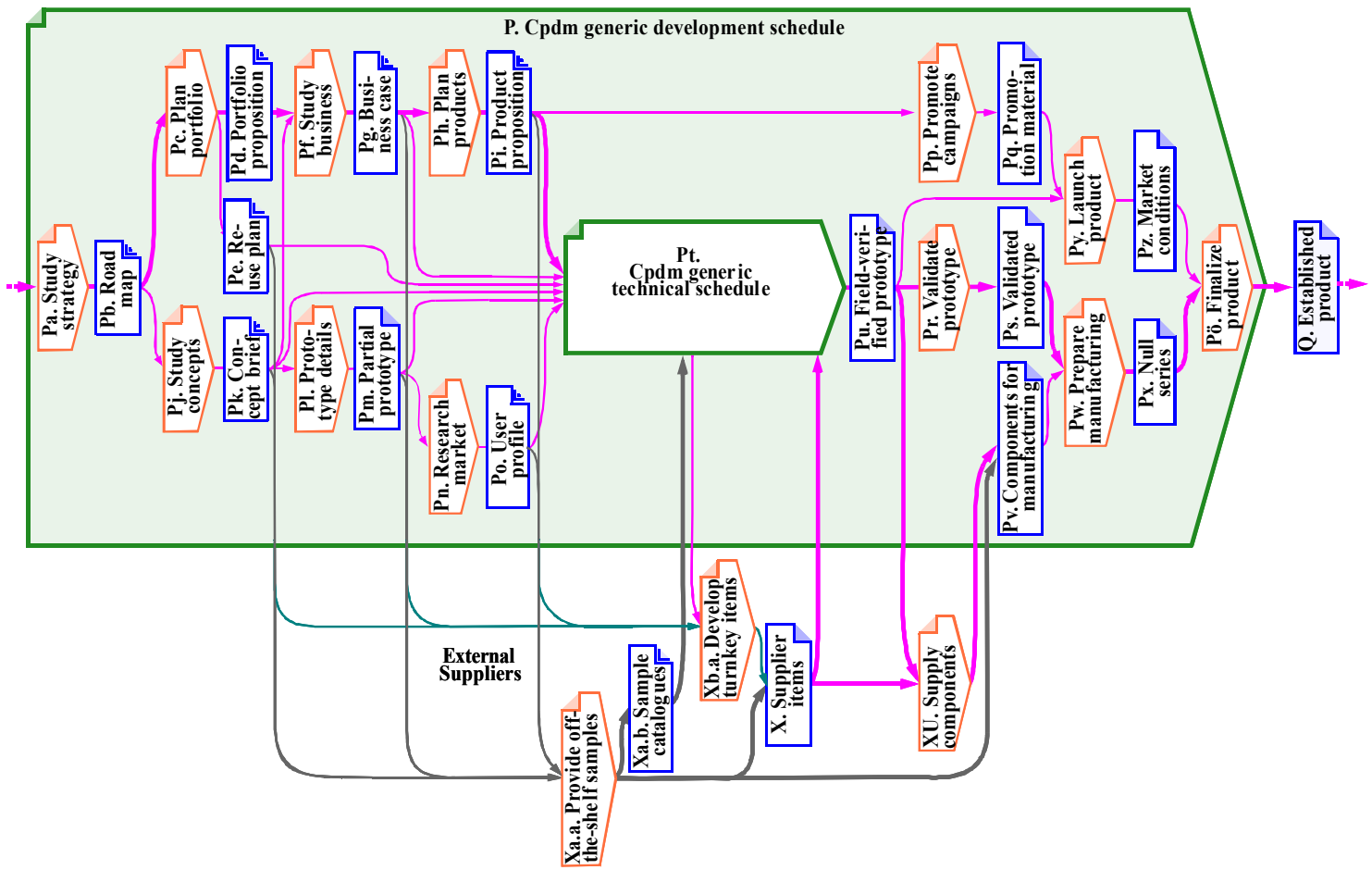
Useful methodologies, Cpdm, own development and manufacturing

- Available worldwide methodologies, including Cpdm
- Own process organization tailor process schedule
- Own developing organization use tailored schedule
- Provided manufacturing use design documentation



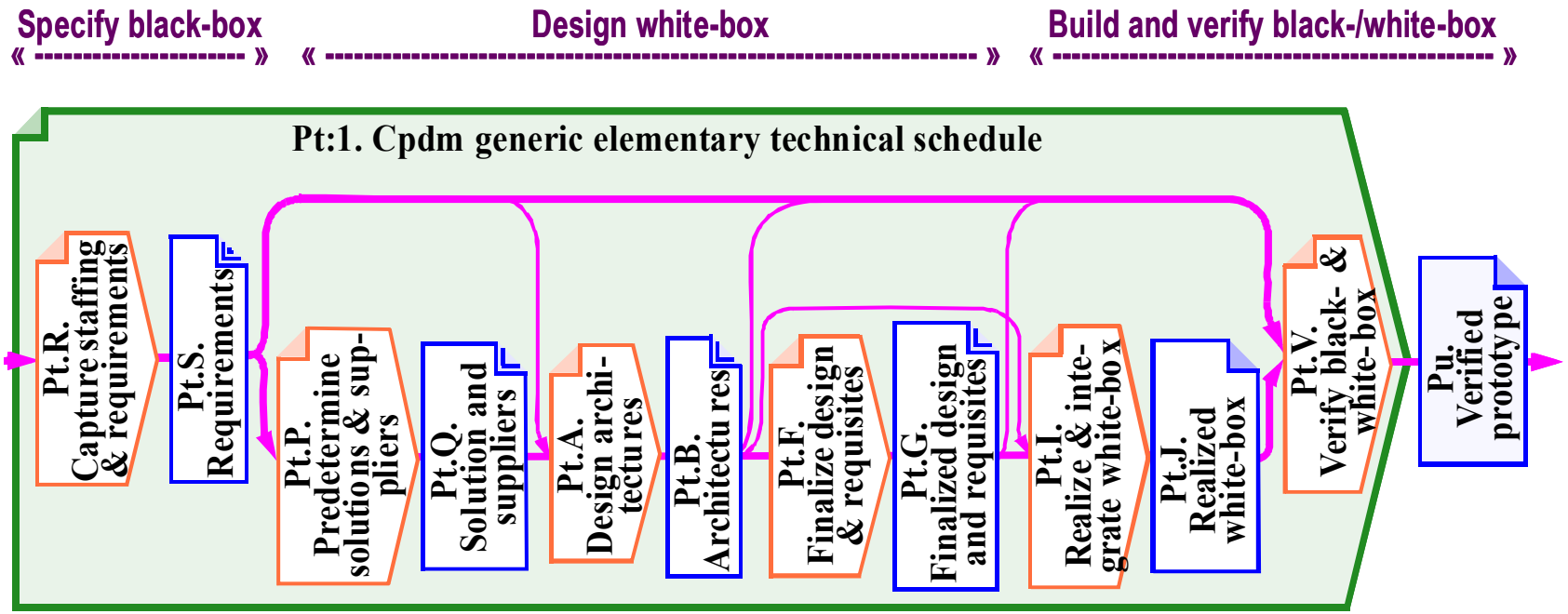
P. Cpdm generic development schedule

- Schedule (P.) for both market and technical development
- From idea to established product
- Suppliers included



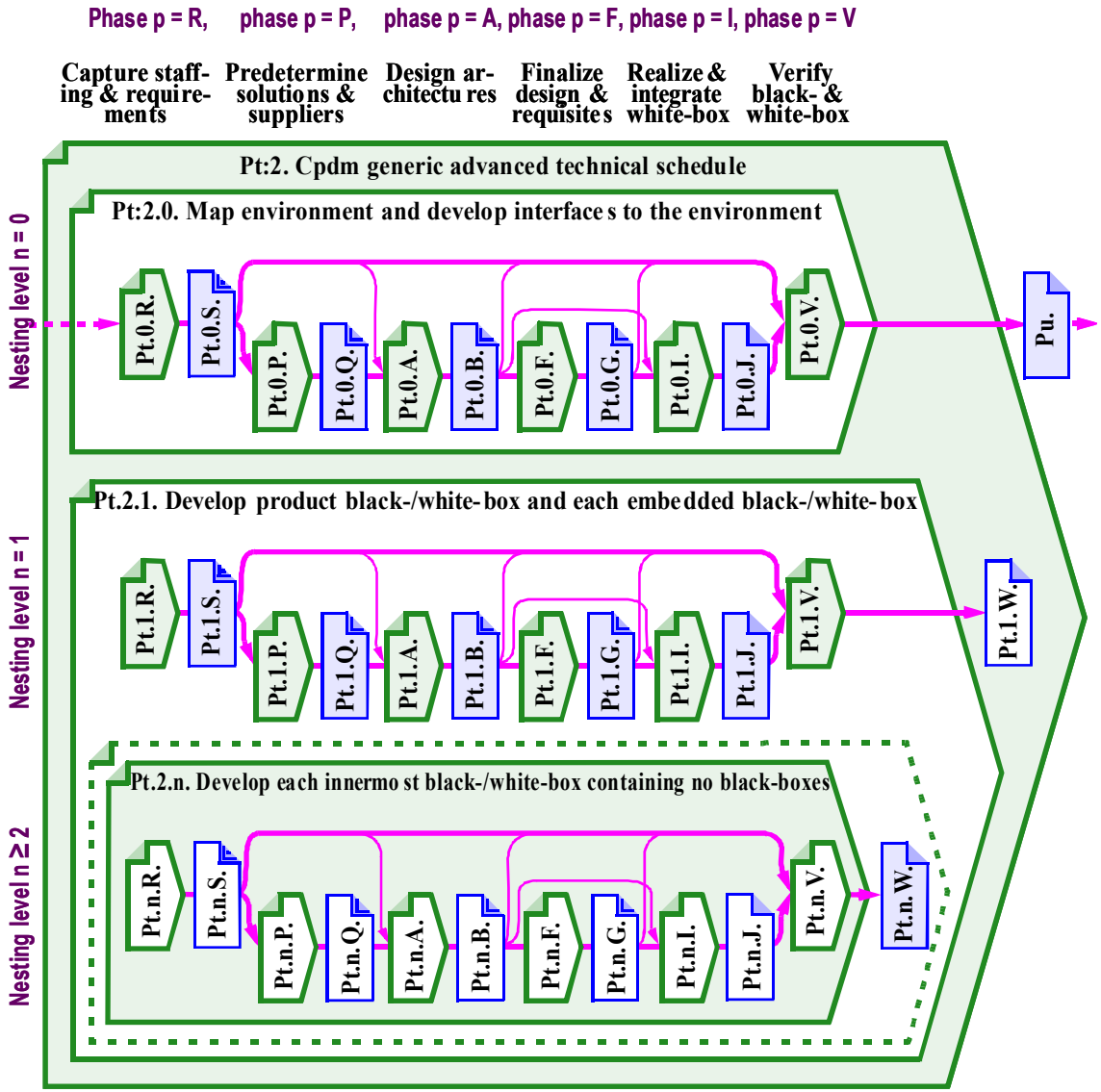
Pt:1. Cpdm generic elementary technical schedule

- Version 1 (Pt:1.) of technical schedule
- For modest complex products



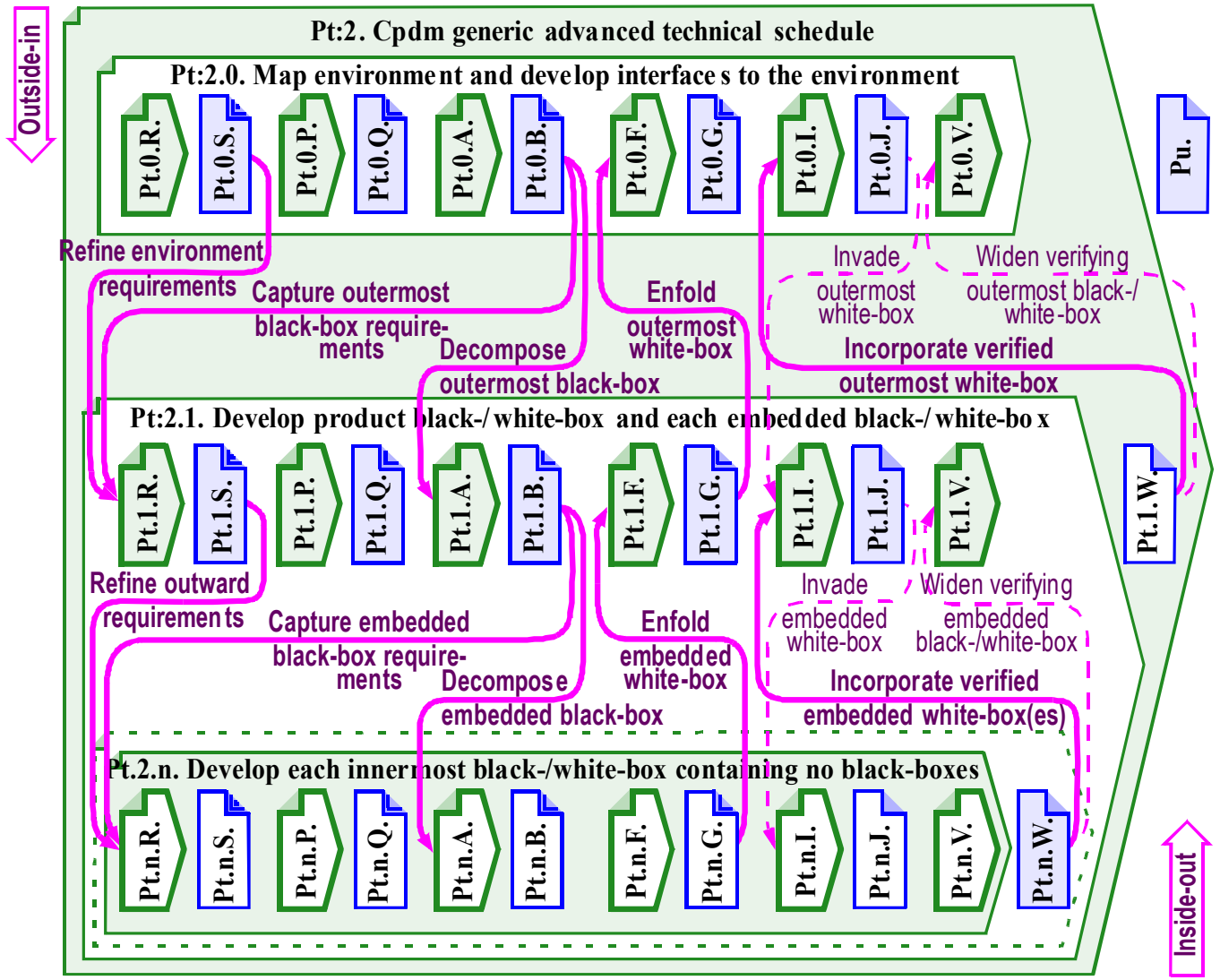
Scaling up technical schedule

- Scale up process according to product architecture hierarchy
- Environment level n = 0 is the surrounding to product
- Product nesting level n = 1 is the product to be developed (including interfaces to environment)
- Product inwards nesting levels n > 1 are hierarchical parts inside the product



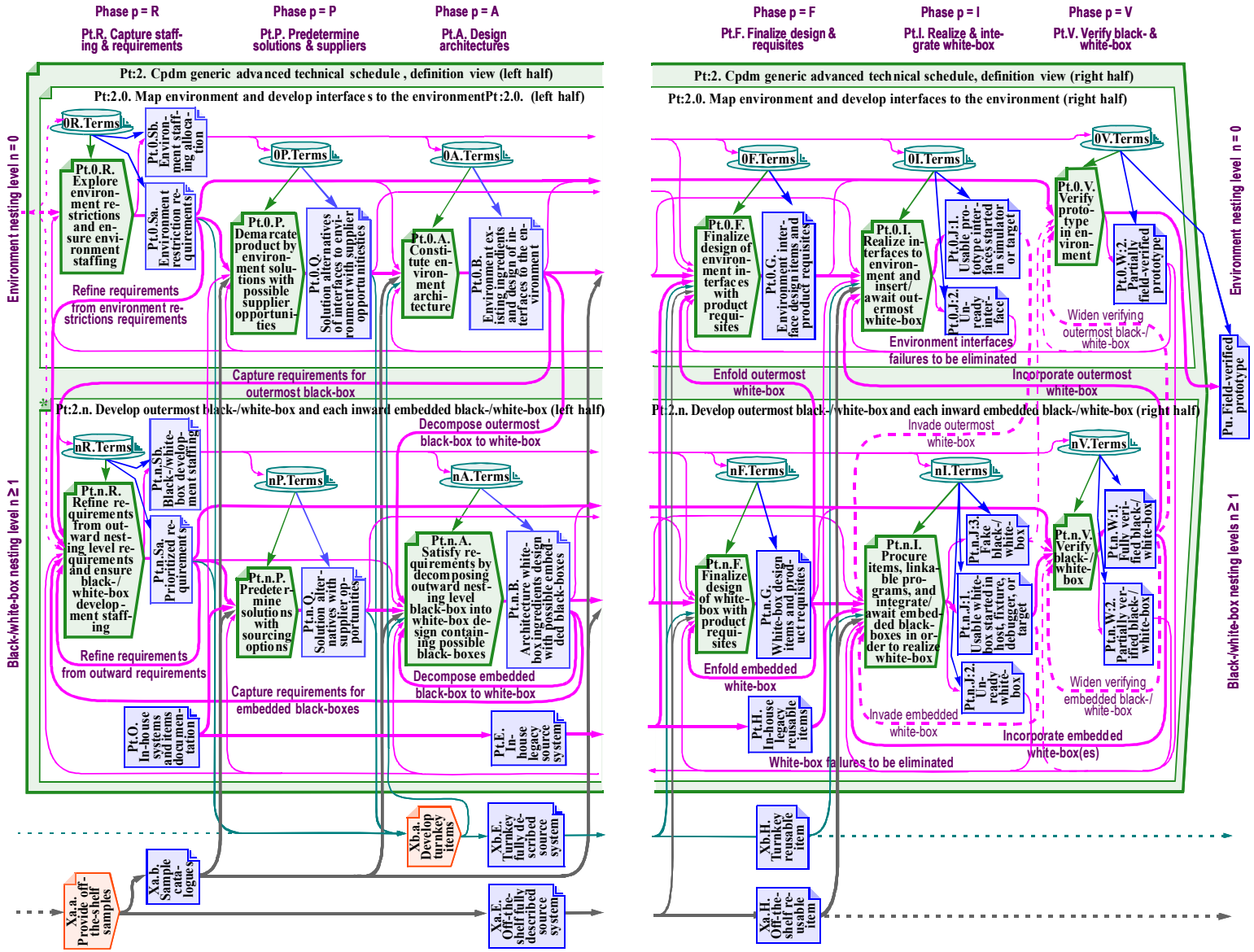
References between levels

- Outside-in is moving from the environment and inwards
- Inside-out is moving from embedded parts outwards to environment
- Two ways of integration
  - Inside-out integration and explicit verification
  - Outside-in invasion and inside-out implicit verification



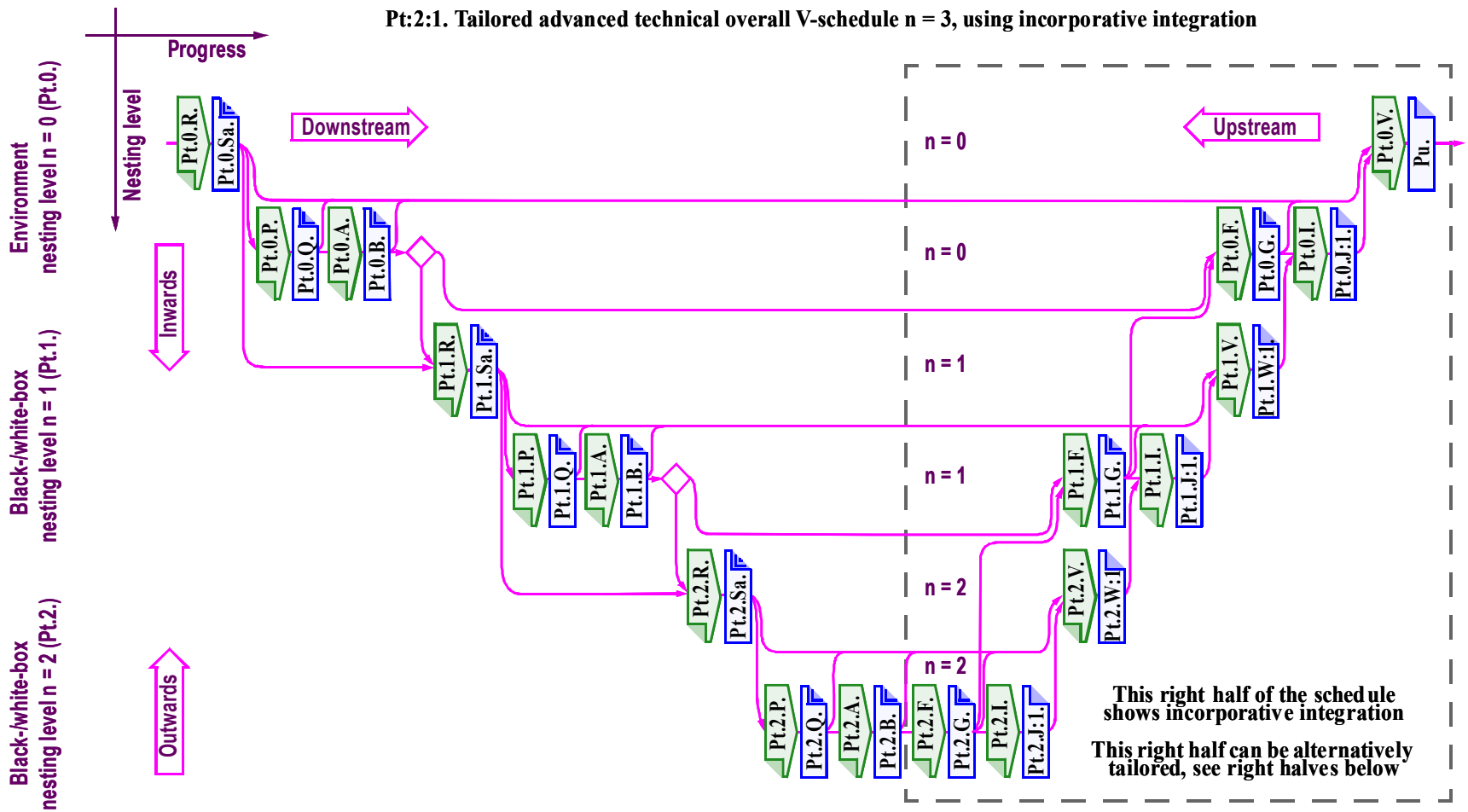
Pt:2 Generic advanced technical schedule

- Version 2 (Pt:2.) of technical schedule
- For definitely complex products

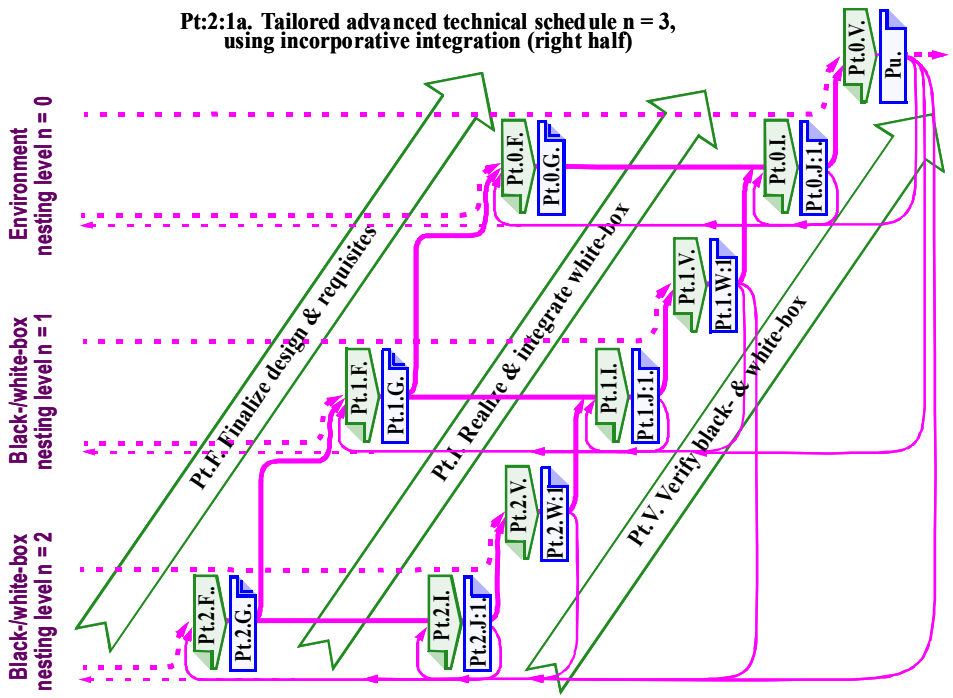


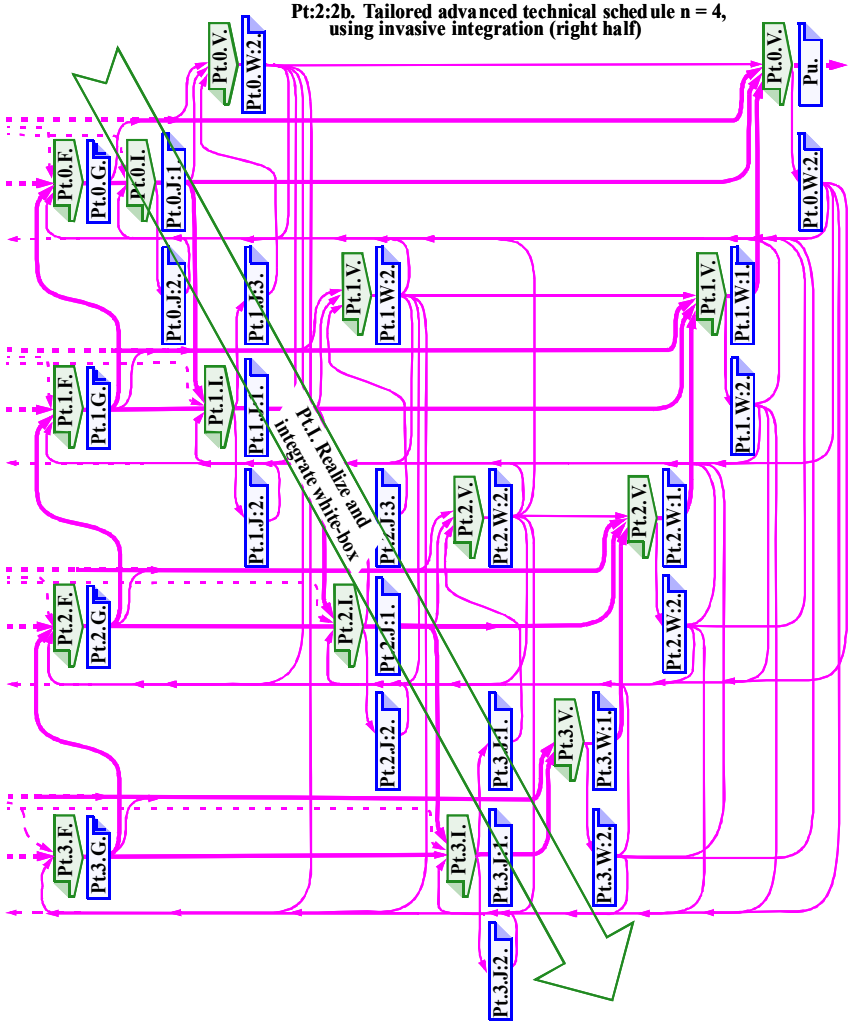
Tailoring the process to V-model

- To show progress
- To show fixed nesting levels n ≤ 3
- To show incorpo-rative inte-gration



Incorporative versus invasive integration





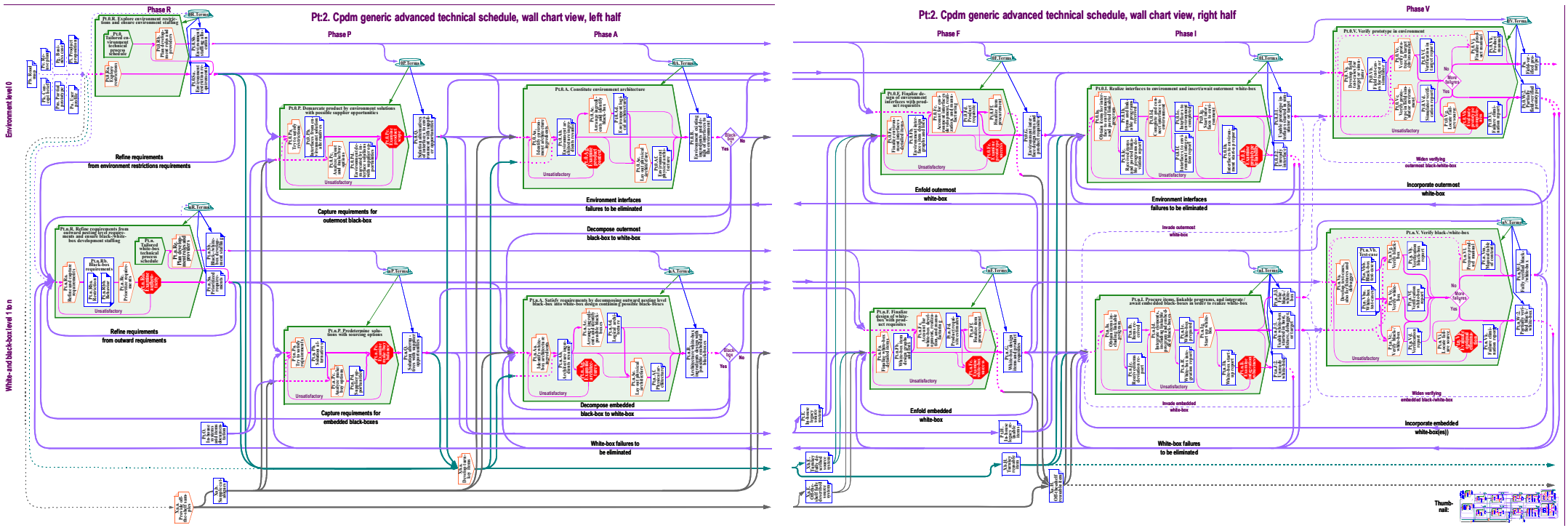
- Sometimes Pt.I. ”Realize and integrate white-box” is prefered executed outside-in.
  - If the white-boxes can hardly not be separately built
  - If the product will be outside-in incrementally validated

Cpdm technical walkthrough with examples

for technical managers and developers

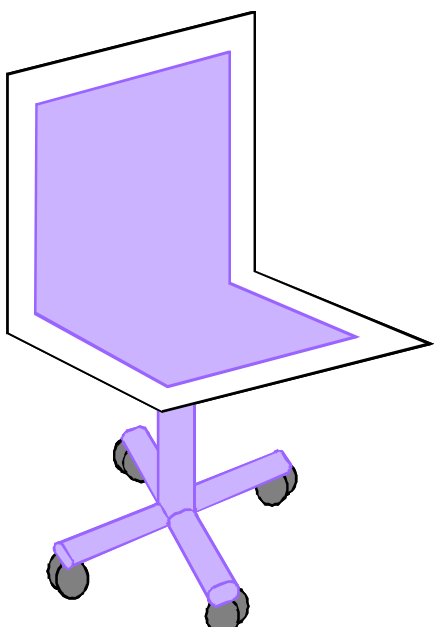
Pt:2 Generic advanced technical schedule wall chart (downloadable in high resolution from www.cpdm.com)

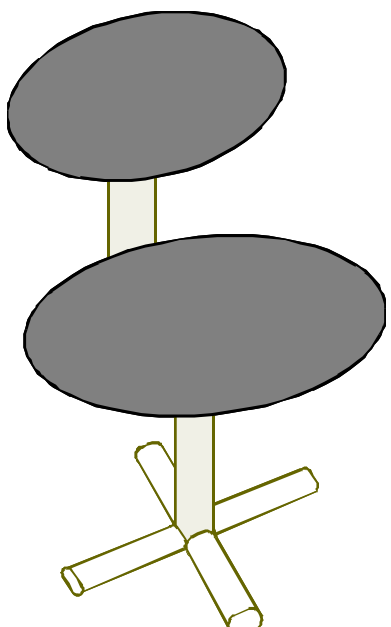
- Used to facilitate navigation within the advanced technical process



EXAMPLE Chairs:
From idea to established product

- Chairs product proposition

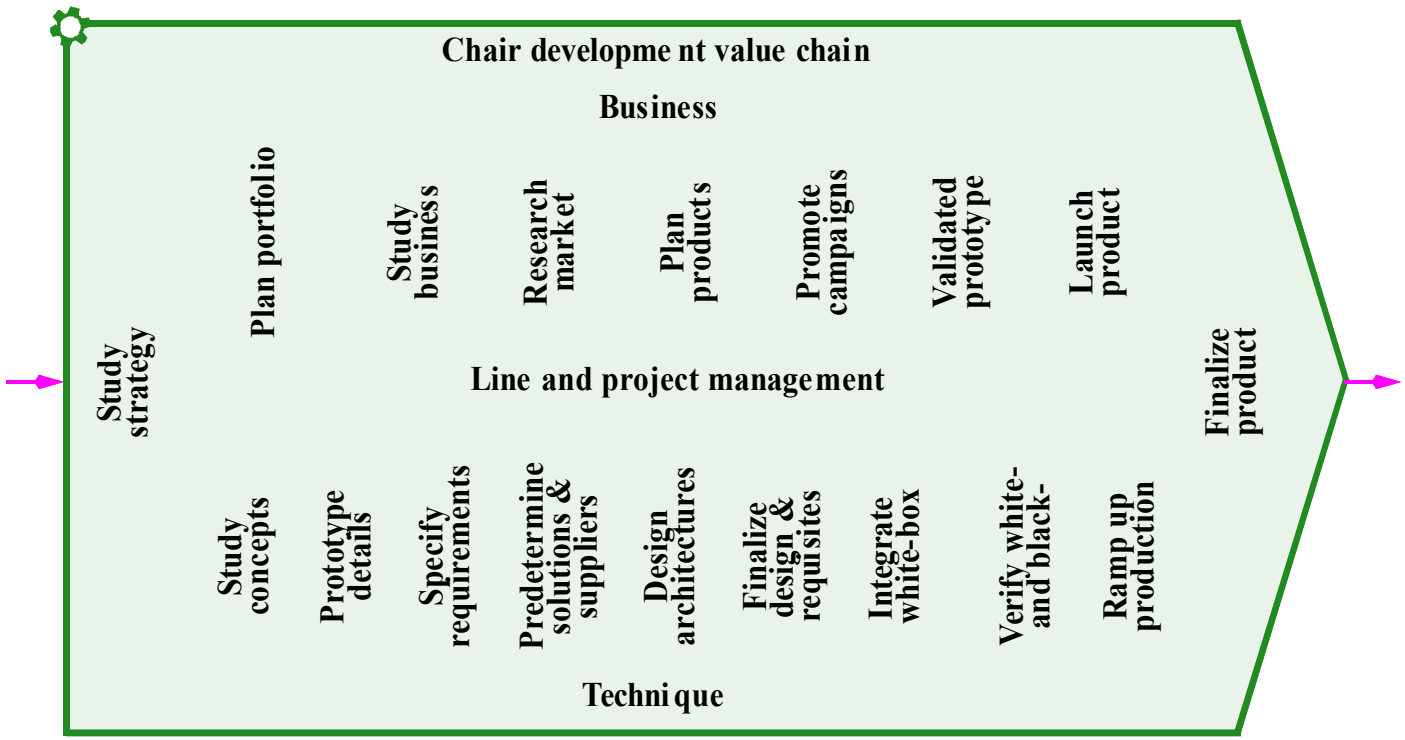




- Chairs concept brief

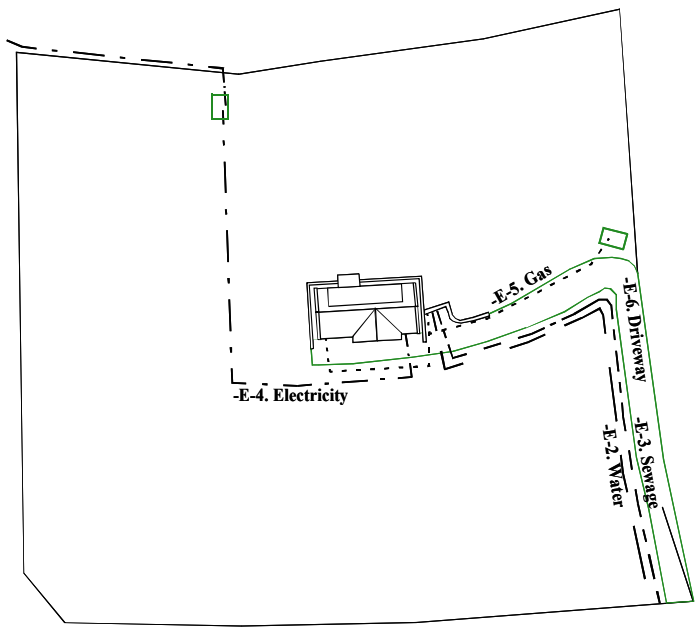
Value chain behind development of chairs

- Chairs modest complexity need only informal value chain
- Value chain not yet captured as documented process



EXAMPLE House:
From specification to verification

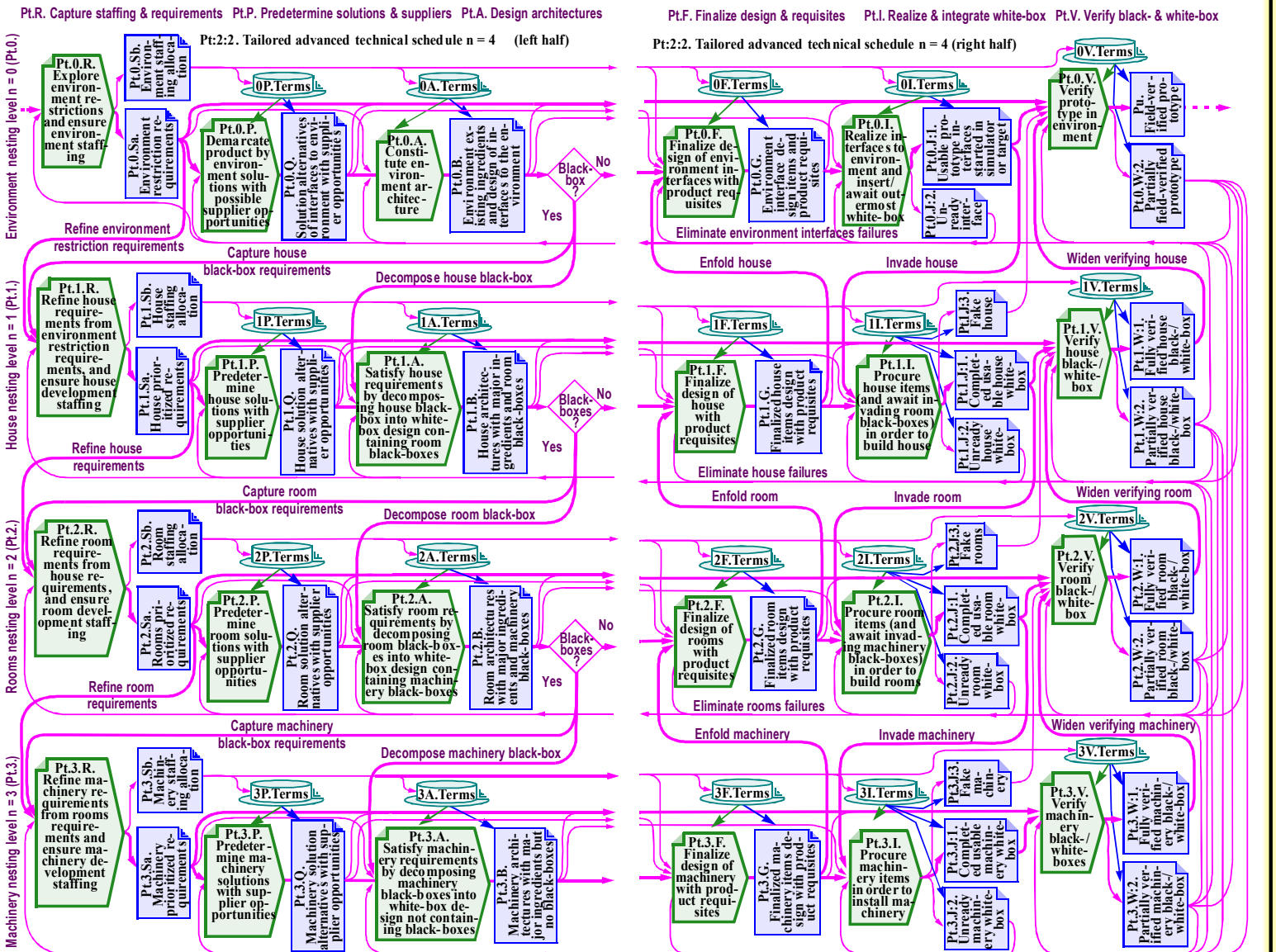
- House environment physical architecture



- House façade physical architecture

Process tailored for house development

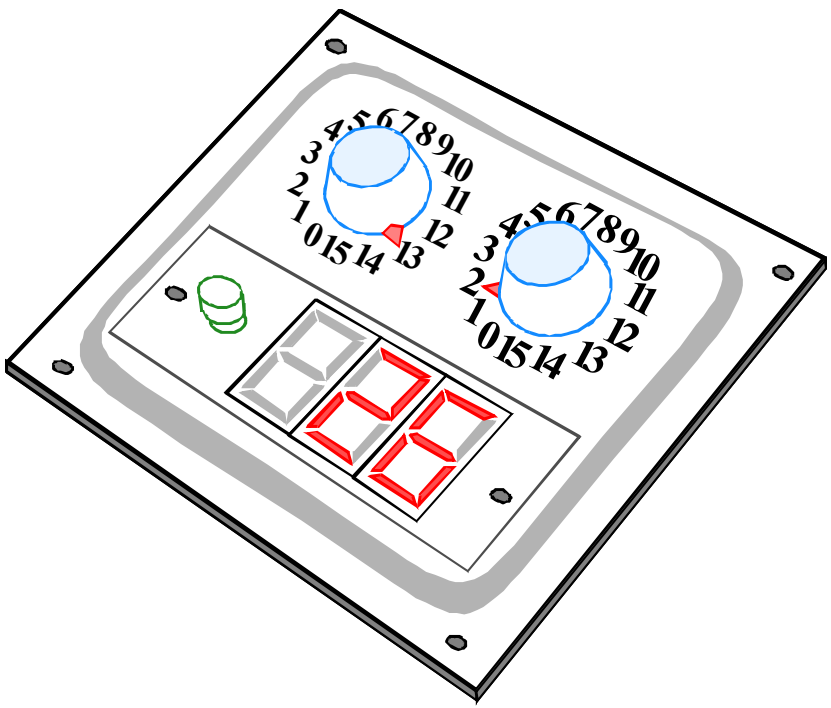
- Tailored advanced technical 4-level schedule (invasive integration)
- High complexity due to large amount of pieces and details



EXAMPLE multiplication toy: Mounted into cuddle toys, realized by 1) Hardwired network and 2) Micro controller

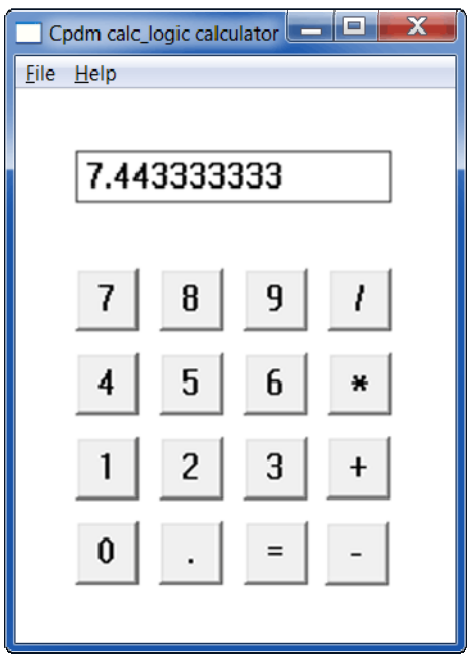
- Physical architecture of environment

- Physical architecture of housing



EXAMPLE Calculator: program component calc_logic ported to 1) Windows application and 2) Pocket device

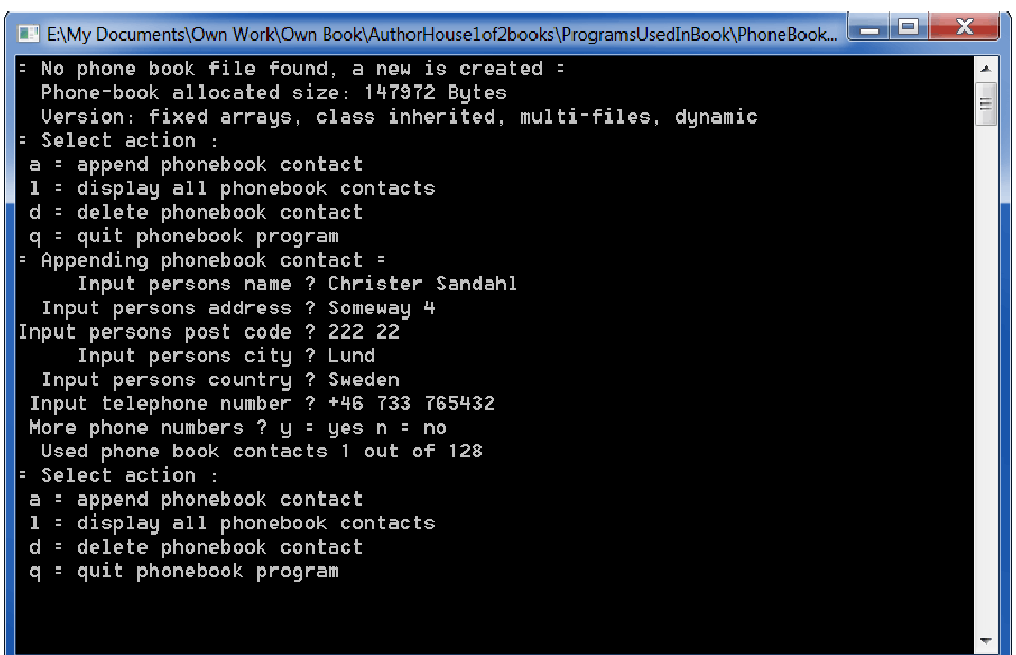
- Calc_logic environment in Windows physical architecture



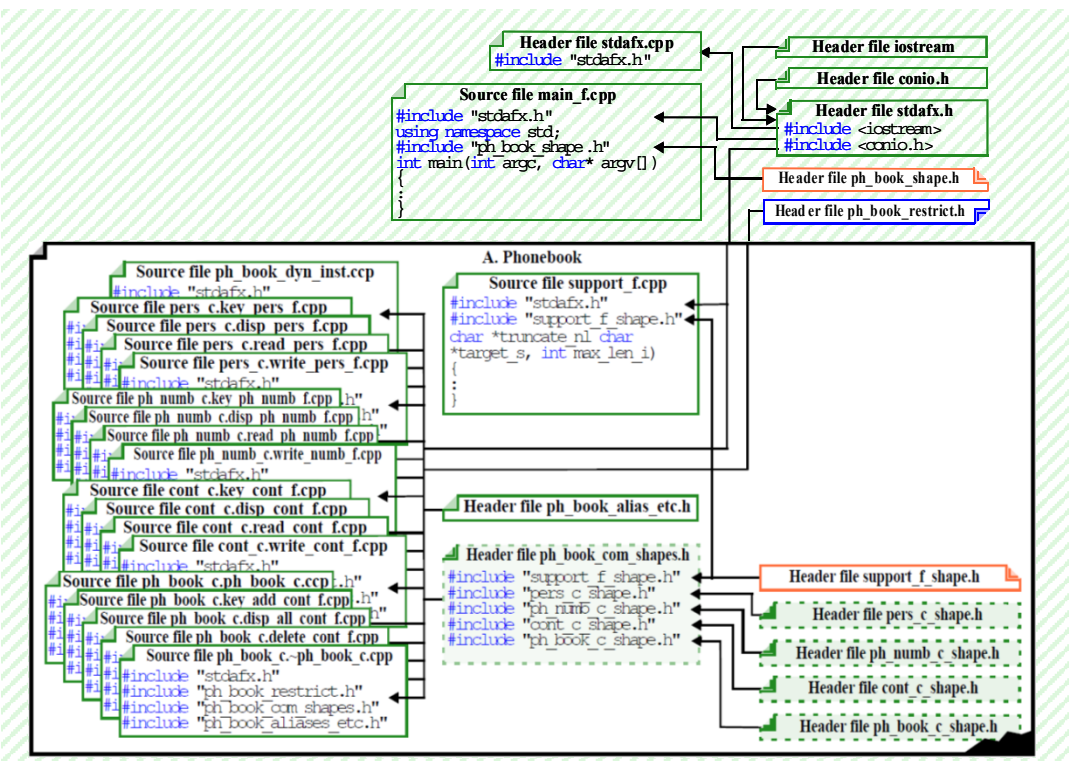
- Calc_logic environment in pocket calculator physical architecture

EXAMPLE Phonebook: Object oriented program

- Phonebook environment physical architecture of mmi

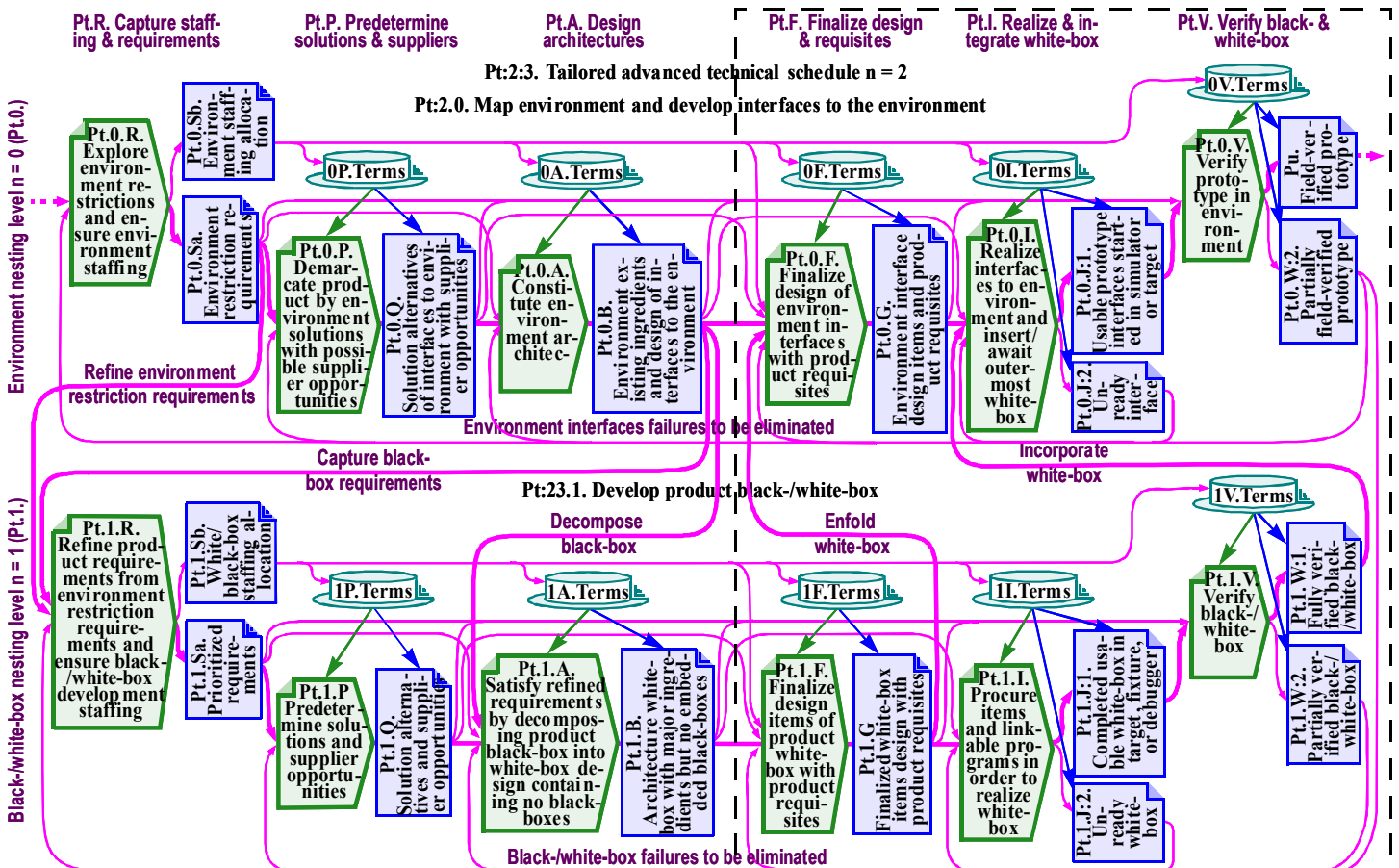


- Phonebook file chart



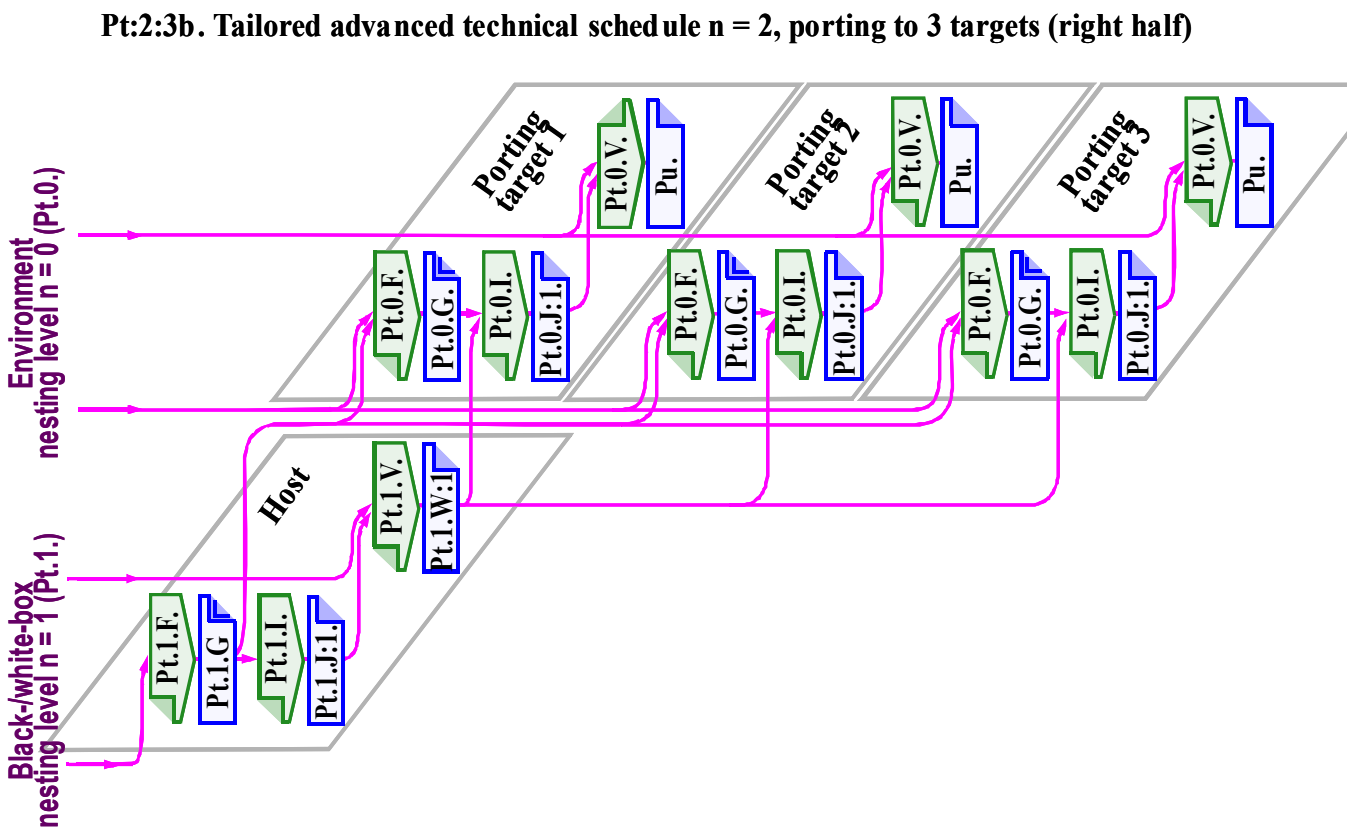
Process tailored for examples multiplication toy, calc-logic reusable component, and phonebook program

- Tailored advanced technical 2-level schedule (incorpo-rative integration)
- Used in examples where product size is not the complexity
- Cpdm simplest advanced technical schedule (only one black-box)



Process tailored for calc_logic example to illustrate porting

- Tailored advanced technical 2-level process V-schedule (incorporative integration)
- Porting to increasingly scarce targets

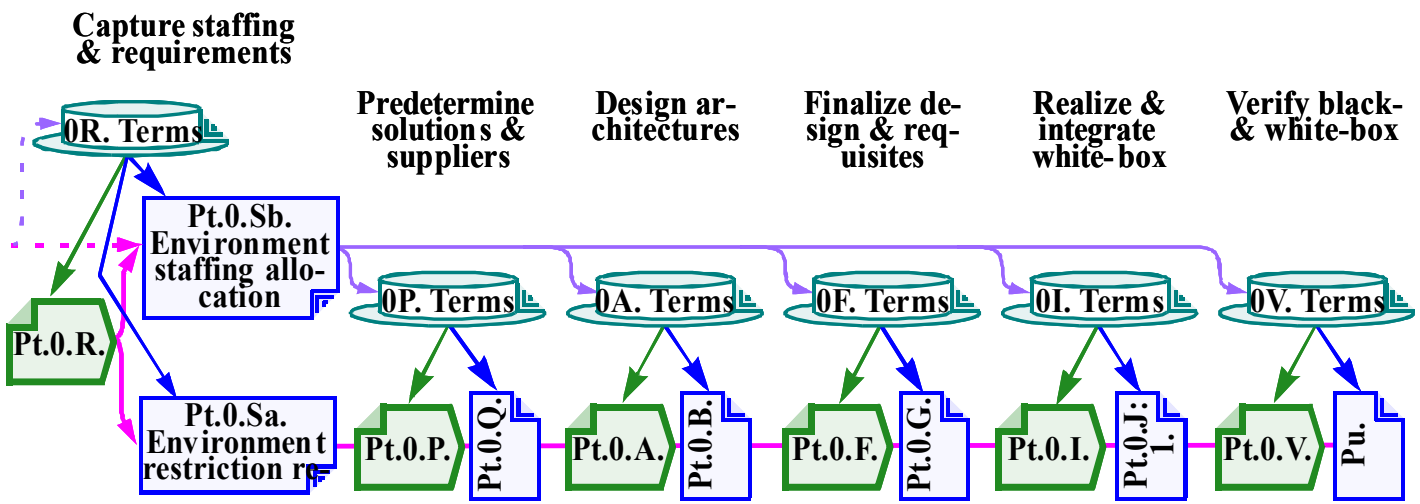


Specify requirements

Getting a bit desperate?

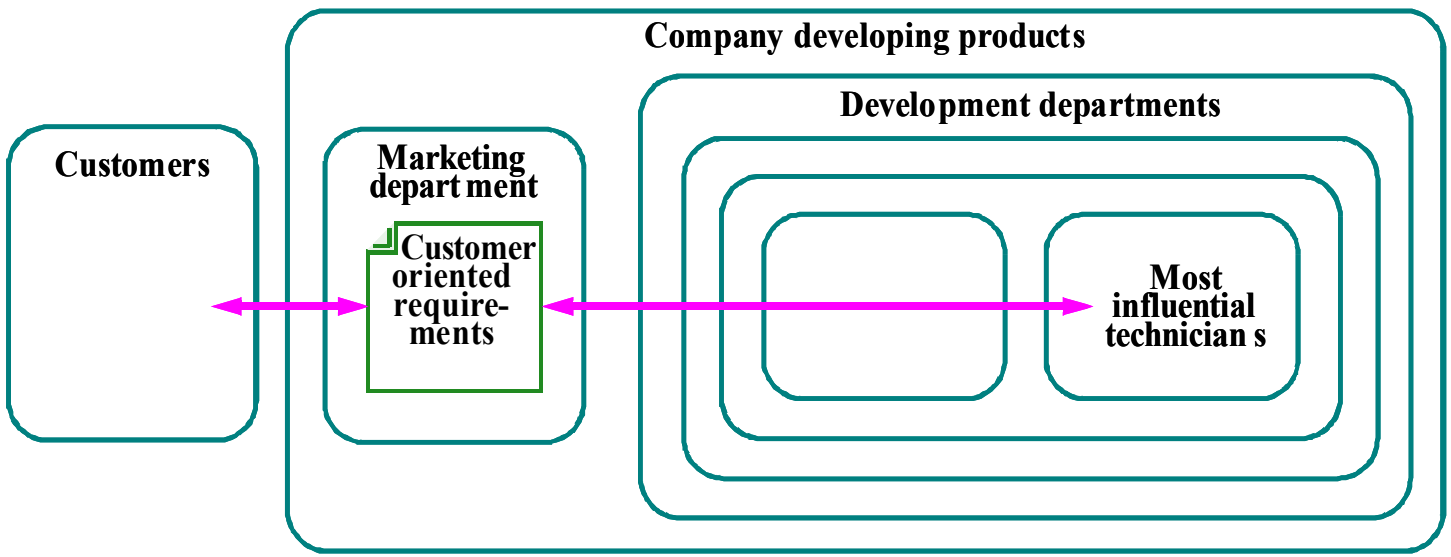
Capture staffing for environment nesting level

- Allocate developers on this nesting level
  - chicken and egg dilemma
- ”Capture staffing & requirement” phase
  - preferably not to be outsourced
  - may need to be started before this phase

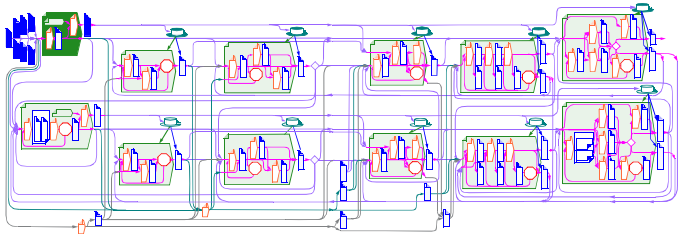


Formalism protects requirements

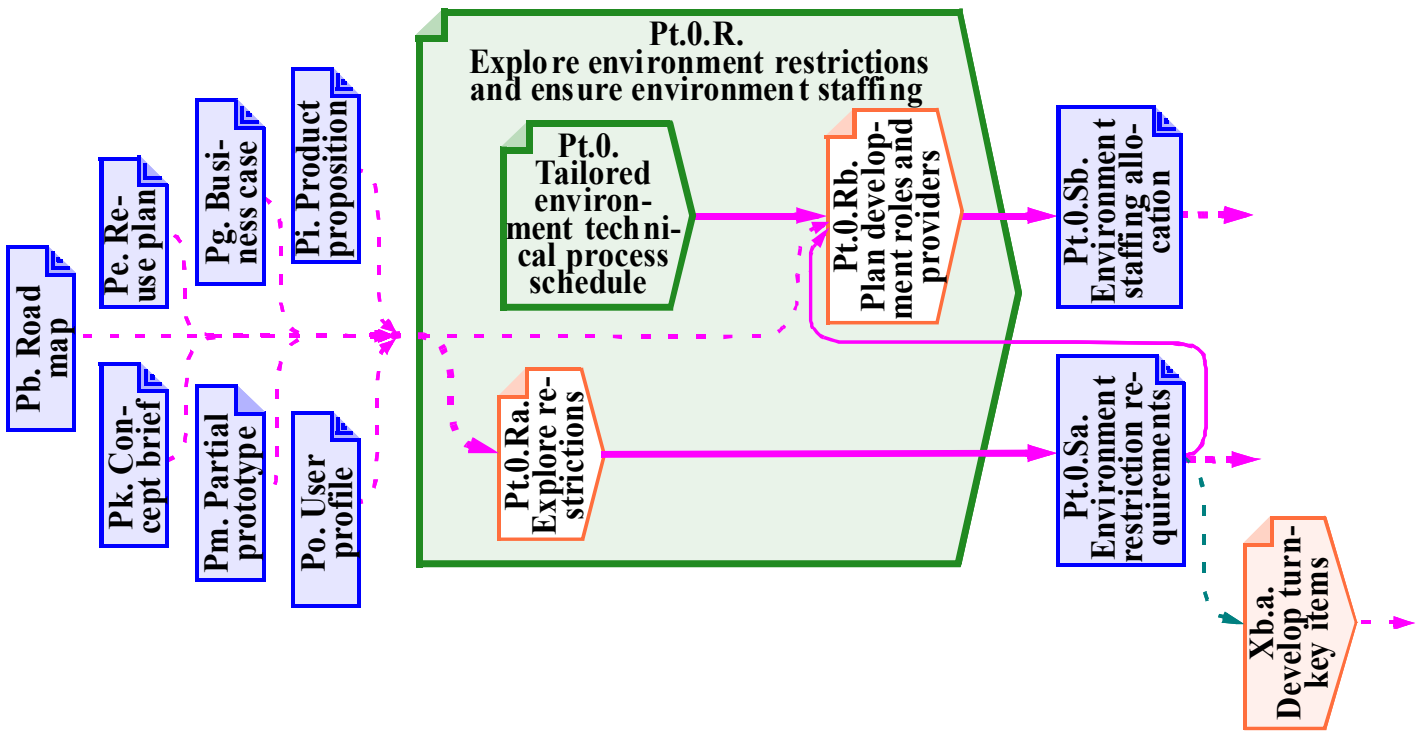
- Convey requirements from customers into core technical development
  - Without requirements being distorted
- To also be used by verification developers



Pt.0.R. Explore environment restrictions and ensure environment staffing



- Develop (preferably inhouse)
  - Tailor a process
  - Specify environment staff allocations
  - Specify environment restrictions
  - Provide restrictions to potential turnkey suppliers



EXAMPLE House: Environment staffing allocations

- Allocate roles to every task and schedule
- Inhouse and third party providers
- Role items will be accounted in product requisites

House environment role item	Provider	Cpdm schedule or task
i10. House user	**PR.10. Proprietor**	Pt.0.Rb Plan development roles and providers
i11. City architect	**PR.11. Robber HB**	Pt.0.Ra Explore restrictions
i12. City architect • For communal connections • Reviewed by proprietor.	**PR.11. Robber HB**	Pt.0.Pa. Satisfy restrictions
i13. House user • For driveway • Reviewed by Robber HB.	**PR.10. Proprietor**	
i14. House user • Assistance from Robber HB.	**PR.10. Proprietor**	Pt.0.Pc. Analyse in/out and make/buy options
i15. Chief architect • Responsibilty by Robber HB to comply to authorities regulations and requirements • Reviewed by proprietor.	**PR.11. Robber HB**	Pt.0.A. Constitute environment architecture Pt.0.F. Finalize design of interface connections to environment
i16. Electrician	**PR.12. Elert AB**	
i17. Excavator with driver and plumber **i18. Plumber** **i19. Excavator with driver and electrician** **i20. Excavator with driver and authorized gas plumber** **i21. Dozer with 6 pavers** • Not yet selected. • The builder may use preferred subcontractors. Builder and preferred subcontractors are responsible to comply to authorities building regulations. • Correction of found failures	**PR.13. Builder**	Pt.0.I. Realize interface and integrate/invade outmost white-box (build the house connections)
i22. Electrician • For electricity • Correction of found failures	**PR.12. Elert AB**	
i23. Quality inspector • Each supplier and subcontractor approve independent quality inspector • Detects and document deviations to requirements and design • Attest invoices for payment when found failures are corrected	**PR.10. Proprietor**	Pt.0.V. Verify prototype in environment
i24. Inspector • Approves building permit and house usage	**PR.14. Commune**	Pt.0.A. Constitute environment architecture Pr. Validate prototype

EXAMPLE Calc_logic: Environment restrictions

- Calc_logic reusable program component have two environments
  - 1) Windows application and 2) Pocket calculator

calc_logic environment (Windows calculator application) environment restrictions

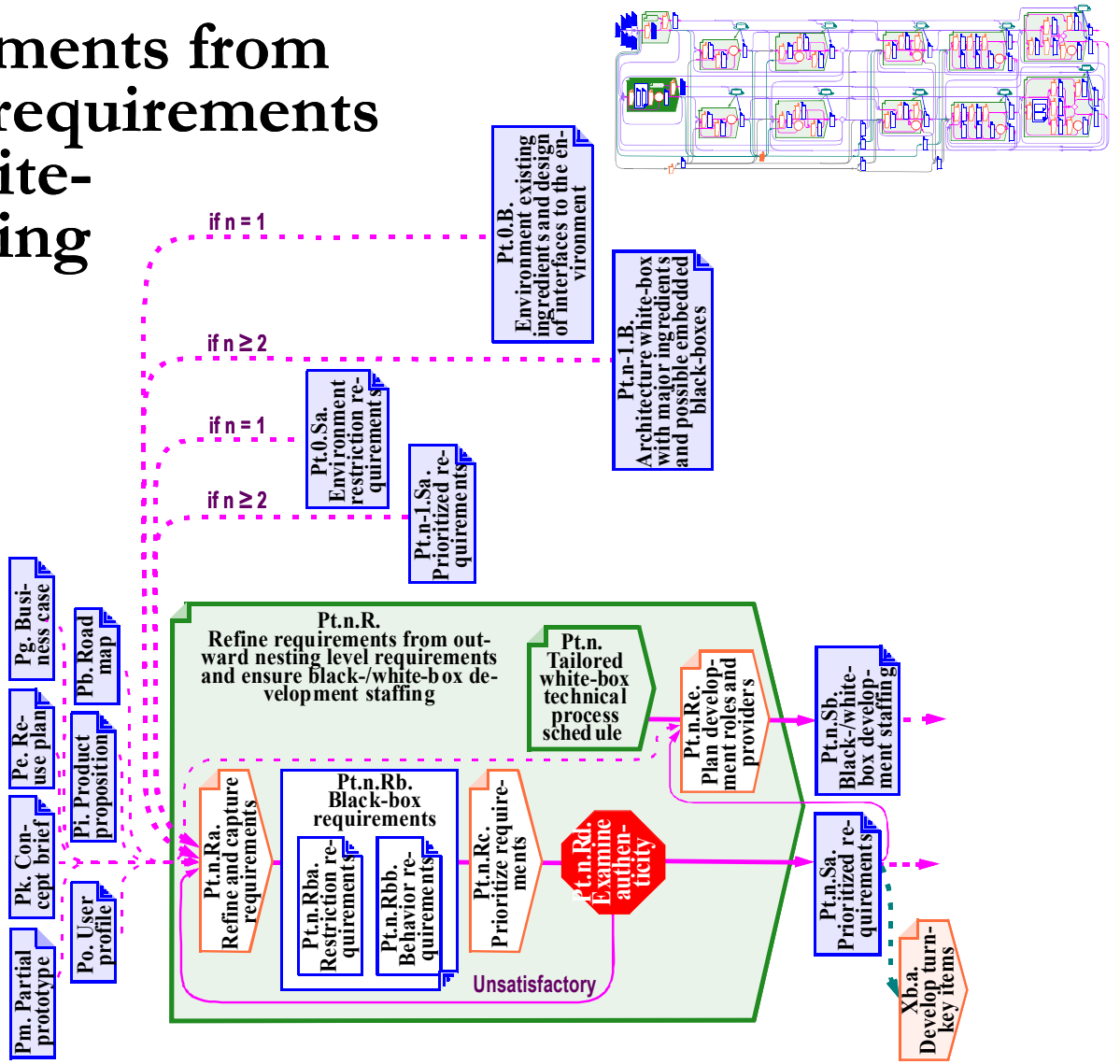
	Environment restriction
E. RR.7.	Since a window screen can hold almost as many buttons as desired, a large amount of buttons shall be possible.
E. RR.8.	Since a window screen field can show almost as many alpha-numeric characters as desired, the floating point precision shall be as high as possible and the display shall be able to show all accurate digits.
E. RR.9.	If a math coprocessor is available in the computer, it may be used to heighten calculation speed. However, the calculating software shall also function in computers without math coprocessor.
E. RR.10.	There is a lot of compilers for windows, having a lot of different kind of mathematical calculation support. Standard Windows C compiler with library supports has float (~ 7 digits), double float (~ 15 digits) and long double float (~ 31 digits) precision. However, the long double variables most often are diminished to same precision as double variables. As Windows 64 bits version gets more popular, many compilers will probably in near future offer real support for long double (~ 31 digits) floating point precision.
E. RR.11.	Many kind of third party mathematical libraries are available or portable to Windows.
E. RR.12.	For proper reusability, calc_logic software must integrate and interface to other software, prepared by most major development systems on Windows

calc_logic environment (pocket calculator firmware) environment restrictions

	Environment restriction
E. RR.1.	To fit market cost of an affordable and everyday calculator, the number of buttons shall be maximum 16.
E. RR.2.	To fit market cost of an affordable calculator, the stand alone calculator display shall hold maximum 8 digits.
E. RR.3.	Micro processors for micro controllers has rarely on-chip or external chip support for mathematical arithmetic.
E. RR.4.	Standard compilers to micro controller seldom have support for extensive mathematical arithmetic, but there are some micro controller compilers with floating point library support for float variables (~ 7 digits).
E. RR.5.	Mathematical software libraries for micro processors can be inhouse developed or outsourced tailor made, and for some micro processors portable libraries are available from third party shelves.
E. RR.6.	For proper reusability, calc_logic software must be able to integrate and interface other software, prepared by compiler and assembler development systems for widely available cheap micro controllers

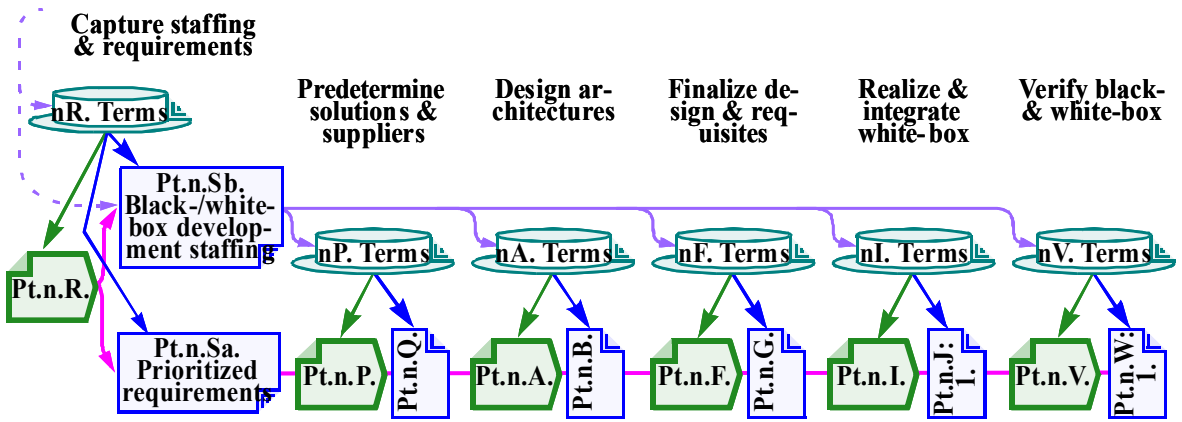
Pt.n.R. Refine requirements from outward nesting level requirements and ensure black-/white-box development staffing

- Develop (preferably inhouse)
  - Tailor a process for this level
  - Specify staff allocations for this level
  - Refine requirements from outward level
  - Prioritize requirements
  - Provide requirements to potential turnkey suppliers



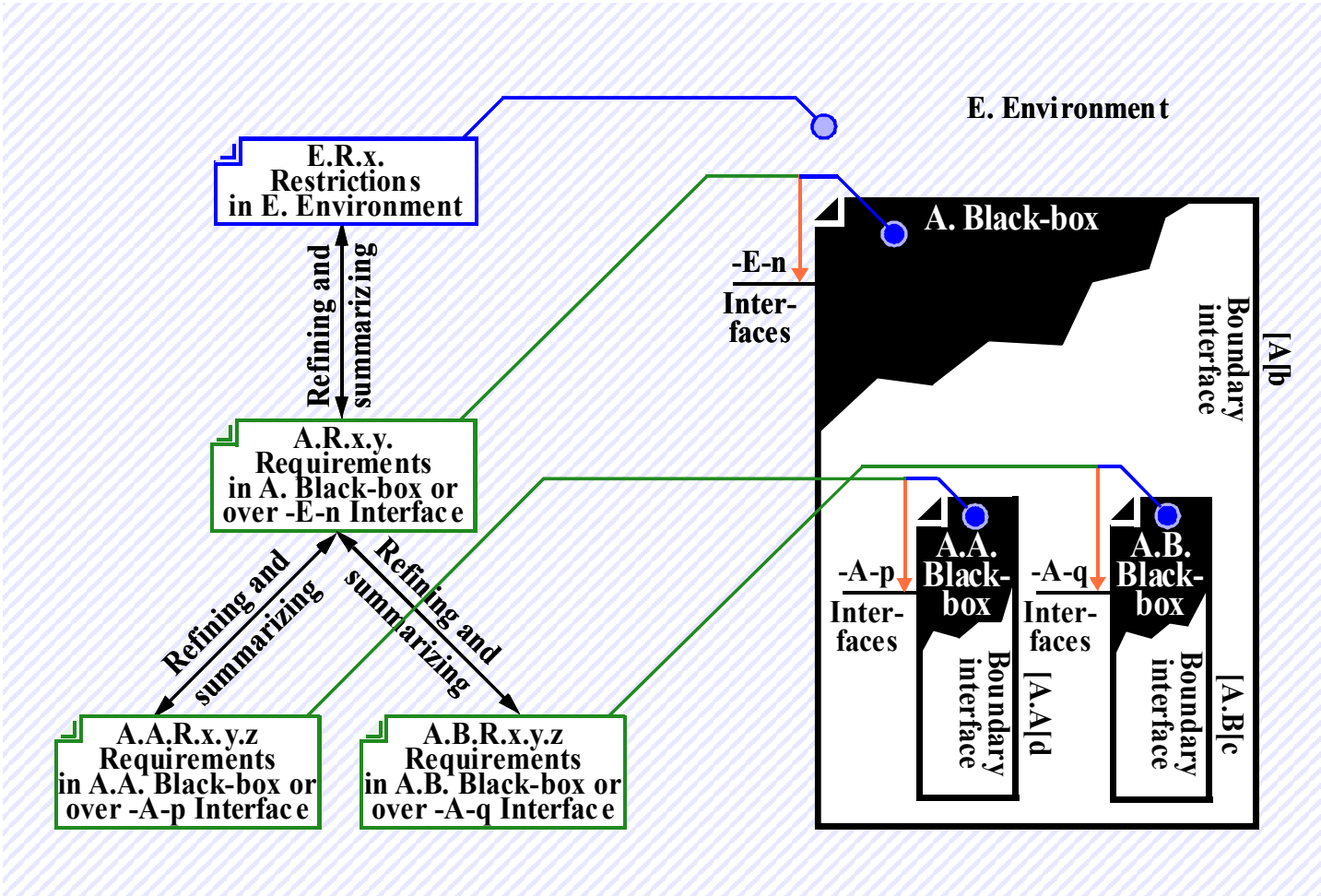
Capture staffing for embedded nesting levels

- Planning of staffing allocation done for the entire nesting level
  - Note that there might be many black-/white-boxes on each nesting level
- “Capture staffing and requirement” phase may be started before entering this phase



Refine requirements from outward black-box

- Environment restrictions are first refined to become outermost black-box requirements
- After all environment restrictions are refined, new requirements may be added for the outermost black-box
- Requirements cover both black-box restrictions requirements and inter-face behavior requirements



EXAMPLE House: kitchen refined require-ments

- House requirements refined to kitchen requirements

House prioritized requirements (small selection)

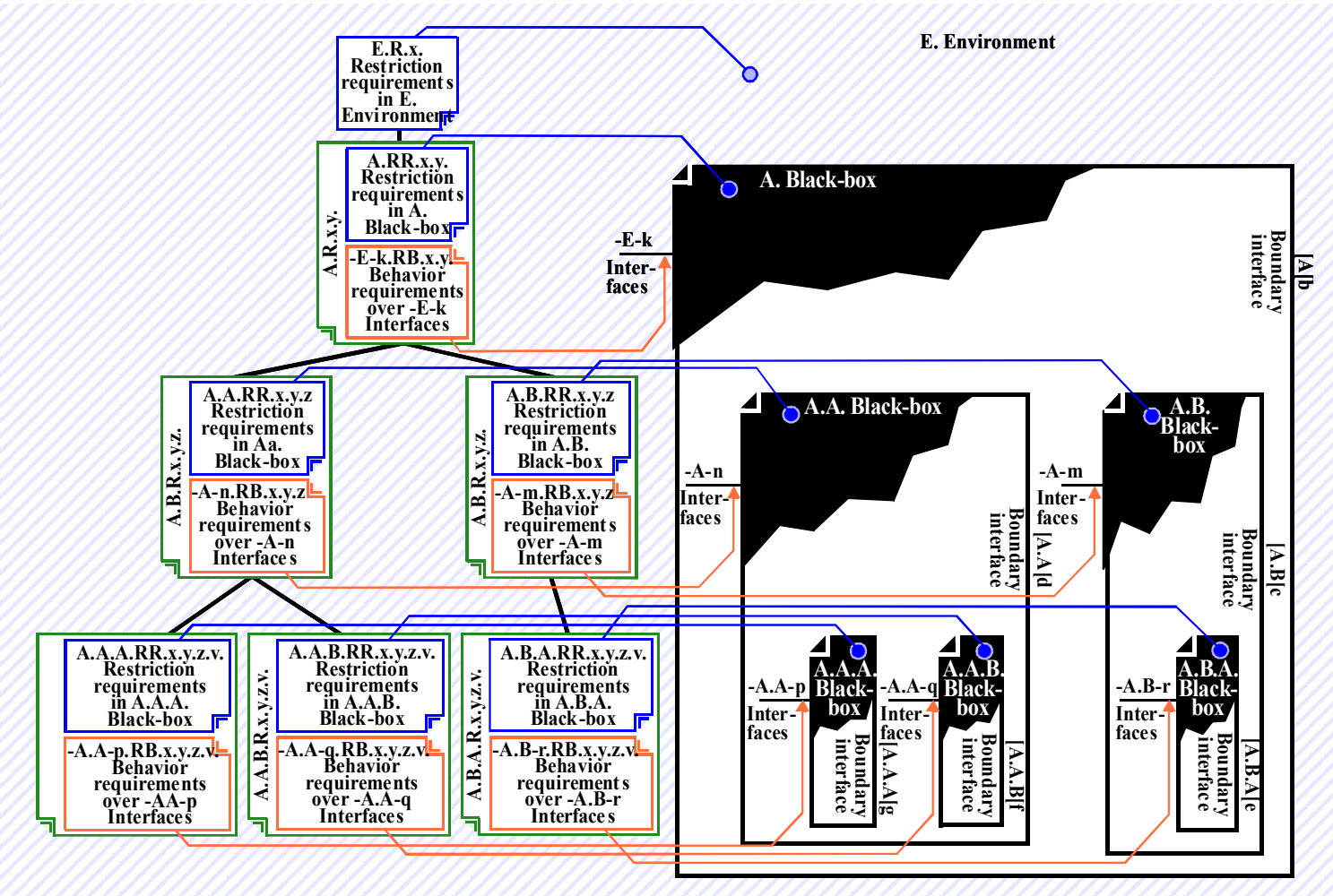
	House black-box requirement	Priority
E. R.1.	Swedish environmental code 7th chapter	
A. R.1.1.	**When cooking and dining, the lake shall be as visible as possible**	**High**
R.1.2.	**Facade and roofing material appearance shall be chosen by proprietor**	**High**
E. R.2.	Malmö urban planning and local construction ordinance	
A. R.2.1.	**The house style shall comply with neighbours' houses.**	**Low**
A. R.2.2.	**The house shall be accessible for disabled persons.**	**Mid**
A. R.2.3.	**The house demand a building permit before building starts**	**Crucial**
E. R.3.	Swedish national building regulations	
A. R.3.1.	**The house shall follow proven methods, regulations and standards**	**High**
R.3.2.	**The house shall have an economical energy consumption**	**High**
E. R.4.	Swedish Financial Supervisory Authority	
A. R.4.1.	**The house shall be attractive to allow maximum bank mortgage**	**Crucial**
E. R.5.	All communal connections available shall be used to make the house functional and comfortable.	
A. R.5.1.	**Provide the house with centrals for all communal connections**	**Crucial**
A. Ch.1.	Modern people in wealthy city districts cook and party	
A. R.6.	Gourmet cooking and dining requirements	
A. R.6.1.	**The atmosphere during dining shall be relaxed.**	**Mid**
A. R.6.2.	**The dining support shall hold up to 8 guests.**	**High**
A. R.6.3.	**Serving support shall exist for one course at a time.**	**Low**
A. R.6.4.	**Storing, cooking and dining support shall allow a french 5 course menu for 8 persons.**	**Mid**
A. R.6.5.	**Dishing support needed, shall hold complete dinner.**	**Crucial**
A. R.6.6.	**Cooking shall support warming, boiling, frying, wokking and baking.**	**Crucial**
A. R.6.7.	**Smell from cooking shall be evacuated.**	**High**
A. R.6.8.	**Waste and garbage from cooking shall be odourless deposited.**	**Mid**
A. R.7.	Reunion and party requirements	
A. R.7.1.	**God space shall exist for 24 party guests, half afoot, half sitting.**	**Mid**
A. R.7.2.	**Serving snack meals shall support 12 party guests.**	**Low**
A. R.7.3.	**Storing capacity shall exist for 24 party guests snack meals.**	**Low**
A. R.7.4.	**Toiletting capacity shall exist for 2 contemporary guests.**	**High**
A. R.7.5.	**Hanger capacity shall exist for 24 outer clothes.**	**Mid**
A. R.8.	Sleeping accommodation requirements	
A. R.8.1.	**(No requirements included here due to book space limitation)**	
A. R.9.	Health and body caring requirements	
A. R.9.1.	**The house shall be safe for children**	**High**
A. Rm.1.	Every house shall be prepared for optic fiber communication	
A. R.10.	Hobbies and electronic communication requirements	
A. R.10.1.	**(No requirements included here due to book space limitation)**	

House kitchen prioritized requirements (small selection)

	House kitchen black-box requirement	Priority
A. R.1.1.	When cooking and dining, the lake shall be as visible as possible	High
A.aC. R.1.1.1.	**The lake shall be visible from kitchen food preparation place**	**High**
A. R.2.2.	The house shall be accessible for disabled persons.	Mid
A.aC. R.2.2.1.	**Kitchen may be inaccessible by disabled guests**	**Mid**
A. R.3.1.	The house shall follow proven methods, regulations and standards	High
A.aC. R.3.1.1.	**Kitchen shall follow established norms, regulations and standards**	**Crucial**
A.aC. R.3.1.2.	**Kitchen interiors may not have any damages, scratches and stains**	**High**
A.aC. R.3.1.3.	**Kitchen interiors appearance must be chosen by proprietor.**	**Crucial**
A. R.4.1.	The house shall be attractive to allow maximum bank mortgage	Crucial
A.aC. R.4.1.1.	**Kitchen including interiors shall cost maximal 20 000 €**	**Crucial**
A.aC. R.4.1.2.	**Purchase only ordinary interiors from few suppliers to press prices**	**High**
A. R.6.3.	Serving support shall exist for one course at a time.	Low
A.aC. R.6.3.1.	**Cheap equipment keep one course warm during eating it**	**Mid**
A. R.6.4.	Storing, cooking and dining support shall allow a french 5 course menu for 8 persons.	Mid
A.aC. R.6.4.1.	**Use large food preparation area with cold and warm water**	**High**
A.aC. R.6.4.2.	**Storage shall hold 12 pans of 3 litre near cooking place.**	**High**
A.aC. R.6.4.3.	**Storage shall hold 6 plates and 6 bowls near baking place**	**High**
A.aC. R.6.4.4.	**Knife storage shall be children safe and near preparation area.**	**Mid**
A.aC. R.6.4.5.	**Storage shall hold all kinds of cutlery and kitchen utensils which shall be easy accessible from preparation area.**	**High**
A.aC. R.6.4.6.	**Storage shall hold 8 bags of each 30 litre groceries, whereof 4 chilled and 1 frozen)**	**Crucial**
A. R.6.5.	Dishing support needed, shall hold complete dinner.	Crucial
A.aC. R.6.5.1.	**Dishing shall clean 5 course table wares in one run (5 x 8 plates, cut-lery and glasses), and dishing shall clean 5 course cooking utensils in same run**	**Mid**
A.aC. R.6.5.2.	**Gentle dishing shall support sensitive table wares**	**Crucial**
A.aC. R.6.5.3.	**Children safe storage shall hold chemicals for dishing etc.**	**Low**
A. R.6.6.	Cooking shall support warming, boiling, frying, wokking and baking.	Crucial
A.aC. R.6.6.1.	**Food thawing shall be near preparation area**	**High**
A.aC. R.6.6.2.	**Warming of all kind of food shall be near preparation area**	**High**
A.aC. R.6.6.3.	**Food boiling shall be near preparation area**	**Crucial**
A.aC. R.6.6.4.	**Food frying shall be near preparation area**	**Crucial**
A.aC. R.6.6.5.	**Food wokking shall be near preparation area**	**Mid**
A.aC. R.6.6.6.	**Food baking shall be near preparation area**	**High**
A. R.6.7.	Smell from cooking shall be evacuated.	High
A.aC. R.6.7.1.	**Odour from cooking may not spread to other rooms**	**High**
A. R.6.8.	Waste and garbage from cooking shall be odourless deposited.	Mid
A.aC. R.6.8.1.	**Garbage odour may not be recognisable in kitchen**	**Mid**
A. R.7.3.	Storing capacity shall exist for 24 party guests snack meals.	Low
A.aC. R.7.3.1.	**Storage shall hold 24 snack meals (2 grocery bags)**	**Low**
A. R.9.1.	The house shall be safe for children	High
A.aC. R.9.1.1.	**Kitchen shall hold children safe family medicines**	**Low**

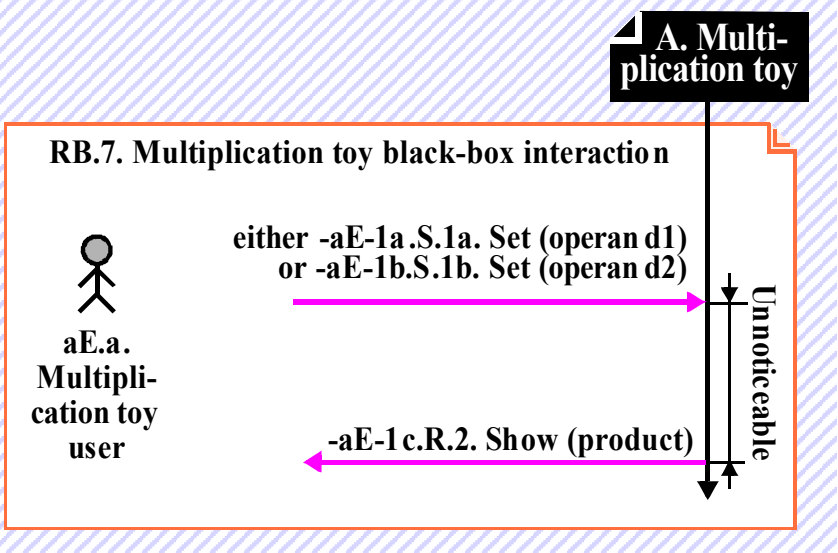
Refine requirements from embedded black-boxes to inward black-boxes

- Requirements divided into 2 types
  - Restrictions requirements to restrain white-box design
  - Behavior requirements on connection interfaces to black-boxes



EXAMPLE Multiplication toy: Behavior requirements

- Tables and graphics



Multiplication toy behaviour requirements

Multiplication toy black-box behaviour interaction		Priority
RB.7.	Multiplication toy black-box interaction see separate figure	
A. RB.7.1.	The user amusement shall be to multiply two operand values into one product value.	Crucial
A. RB.7.2.	The toy shall have one input devices for each multiplicand value. Multiplicand values shall be at least between 0 and 10. The multiplicands must be easy to settle by a child.	0 - 9 crucial > 9 high
A. RB.7.3.	The toy shall have at least a 3-digit output device for the prod-uct value. The product must be easy to read by a child.	Crucial
A. RB.7.4.	The toy calculate and show the product value practically instantly after either multiplicand is settled.	High

Multiplication toy behaviour requirements

Multiplication toy black-box behaviour on interface -aE-1.	
-aE-1a.	Multiplicand input interface
-aE-1a. S.1a.	**Set (operand1)**
-aE-1b.	Multiplicand input interface
-aE-1b. S.1b.	**Set (operand2)**
-aE-1c.	Product output interface
-aE-1c. R.2.	**Show (product)**

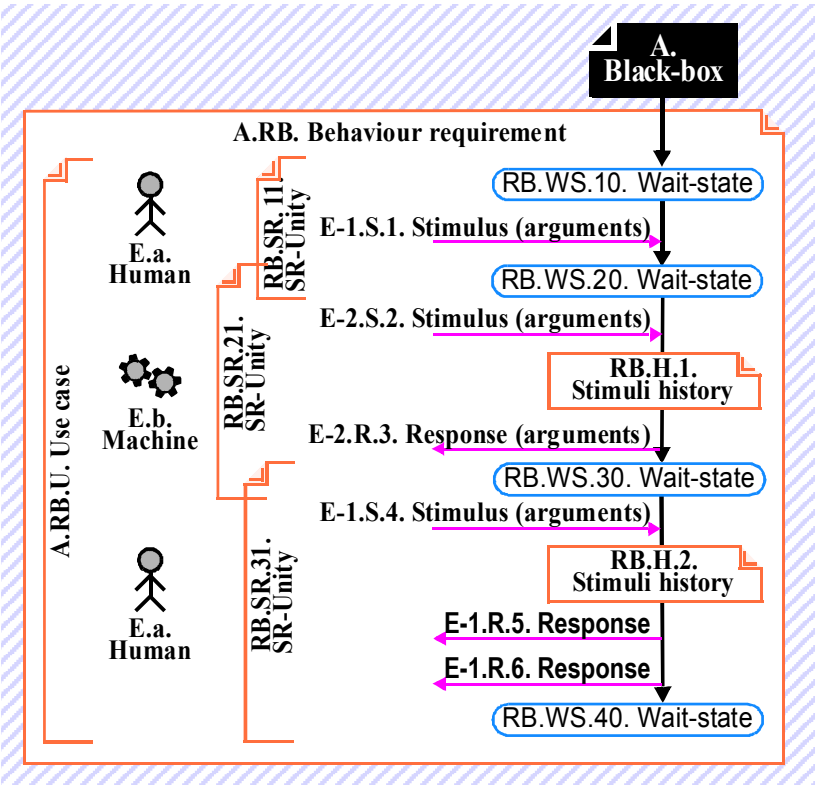
EXAMPLE Rudimentary calculator: Use case

- Basic use case in a table
- Normal text editor can be sufficient
- Normal and exception cases

A.RB.UN. Normal use case		A.RB.UX Exception use cases					
Id, name and interaction		Id, name and interaction		Id, name and interaction		Id, name and interaction	
A.RB.IA.11.	**Input lead operand** The user enters the lead operand by consecutively pressing digits 0-9, back-space and decimal point.	A.RB.IA.12.	**Too many digits entered** Notify user by blinking. Reject digit.	A.RB.IA.13.	**Second decimal point entered** Don't notify user. Reject point.	A.RB.IA.14.	**Backspace when 1 or less digits** Don't notify user.
A.RB.IA.21.	**Input dual operator** The user breaks digit input with one of the dual operators +, -, * or /.						
A.RB.IA.31.	**Input trail operand** The user enters the last operand by consecutively pressing digits 0-9, back-space and decimal point	A.RB.IA.32.	**Too many digits entered** Notify user by blinking. Reject digit.	A.RB.IA.33.	**Second decimal point entered** Don't notify user. Reject point.	A.RB.IA.34.	**Backspace when 1 or less digits** Don't notify user.
A.RB.IA.41.	**Input conclusion.** Conclude with =. Calculate value and show it. Proceed to input new lead operand	A.RB.IA.42.	**Divide by zero** Show error instead of result. Expect new lead operand.	A.RB.IA.43.	**Overflow after + or *** Show value as floating point instead.	A.RB.IA.44.	**Lost significance after small / big** Show result 0. Don't notify user.

Formal use cases

- Table or graphical
- Must contain
  - Identities
  - Interfaces
  - Wait-states
  - Stimuli
  - Responses
  - Histories

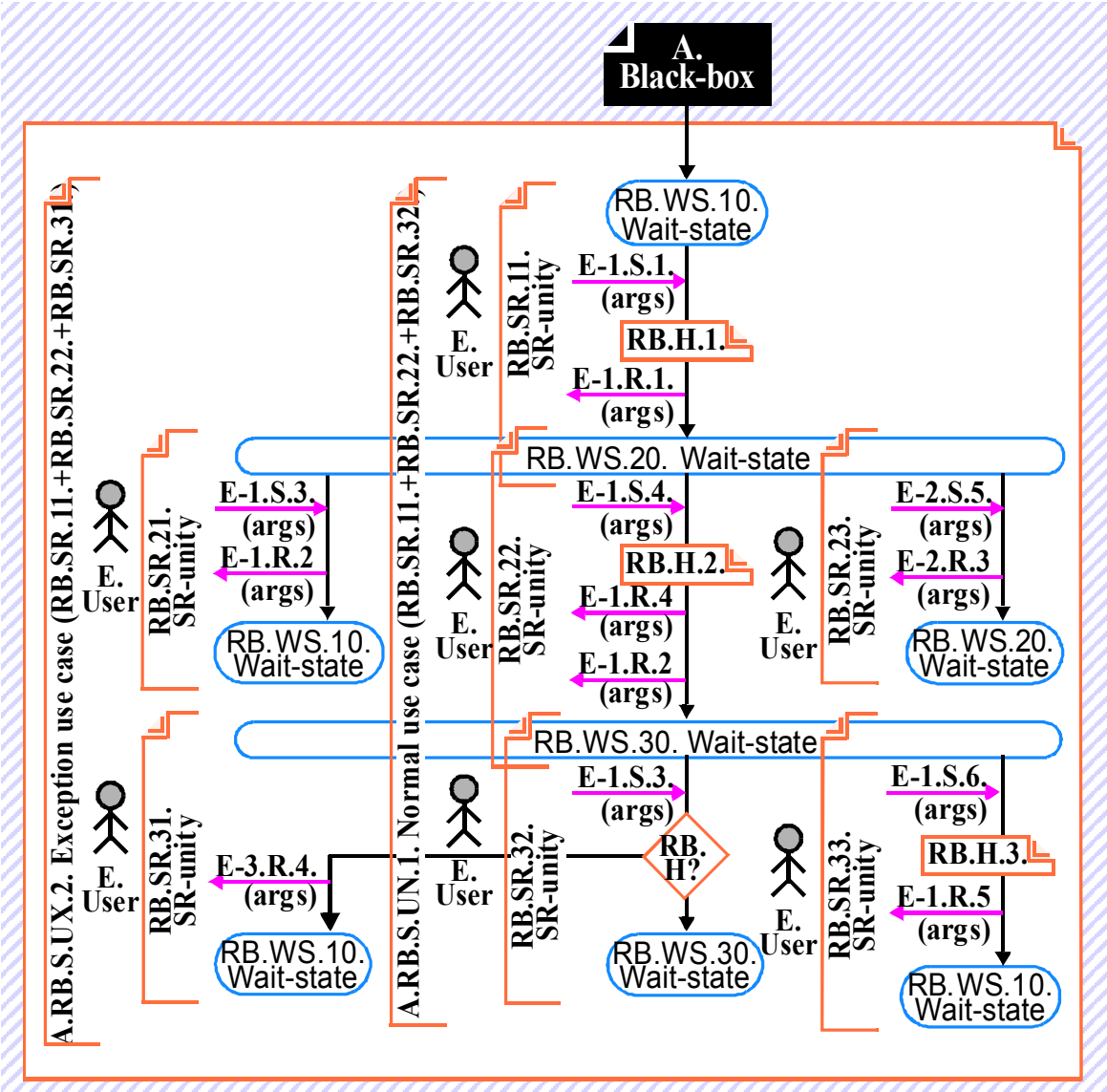


A.RB. Behaviour requirement

A.RB.U Use case	A. Black-box				
	RB.WS.10.Wait-state				
E.a. Human	RB.SR.11.	SR-Unity 11 description	E-1.S-1. Stimulus (arguments)		
	RB.WS.20. Wait-state				
E.b. Machine	RB.SR.21.	SR-Unity 21 description	E-2.S-2. Stimulus (arguments)	E-2.R3. Response (arguments)	RB.H.1. History
	RB.WS.30. Wait-state				
E.a. Human	RB.SR.31.	SR-Unity 31 description	E-1.S-4. Stimulus (arguments)	E-1.R5. Response E-1.R6. Response	RB.H.2. History
	RB.WS.40. Wait-state				

Use case scenario

- Use cases progressing along wait-states
- Initiates preferably in idle-state and can show behavior paths until back to idle-state
- Mixed normal and exception cases
- Can split on many pages



EXAMPLE calc_logic: Behavior in tables (continues)

calc_logic for reuse behaviour requirements

Black-box behaviour use cases		Priority
RB.6.	c alc_logic for reuse behaviour requirement no 6	
A. RB.6.S1.	**calc_logic for reuse, behaviour requirement 6, scenario 1** see separate figure	
A. RB.6.S1.U1.	**Input of lead operand normal use case** sequence of RB.SR.11. + n * RB.SR.21.	**Critical**
A. RB.6.S1.U2.	**Input of trail operand normal use case** sequence of RB.SR.31. + n * RB.SR.41.	**Critical**
A. RB.6.S1.U3.	**Perform completing use case** sequence of RB.SR.11. + n * RB.SR.21. + RB.SR.22. + RB.SR.31. + n * RB.SR.41. + RB.SR.42.	**Critical**
A. RB.6.S1.U4.	**Perform repeating use case** sequence of RB.SR.11. + n * RB.SR.21. + RB.SR.22. + RB.SR.31. + n * RB.SR.41. + RB.SR.43. + ...	**High**
A. RB.6.S1.U5.	**Perform repeating after conclusion use case** sequence of RB.SR.12. + RB.SR.31. + ...	**Critical**

calc_logic for reuse behaviour requirements

Black-box behaviour at interface -E-1.

-E-1. Comm on calc_logic interface	
-E-1. S.1.	**Stim (0 - 9)**
-E-1. S.2.	**Stim (+, -, * or /)**
-E-1. S.3.	**Stim (=)**
-E-1. R.4.	**Resp (lead-operand-story)**
-E-1. R.5.	**Resp (trail-operand-story)**

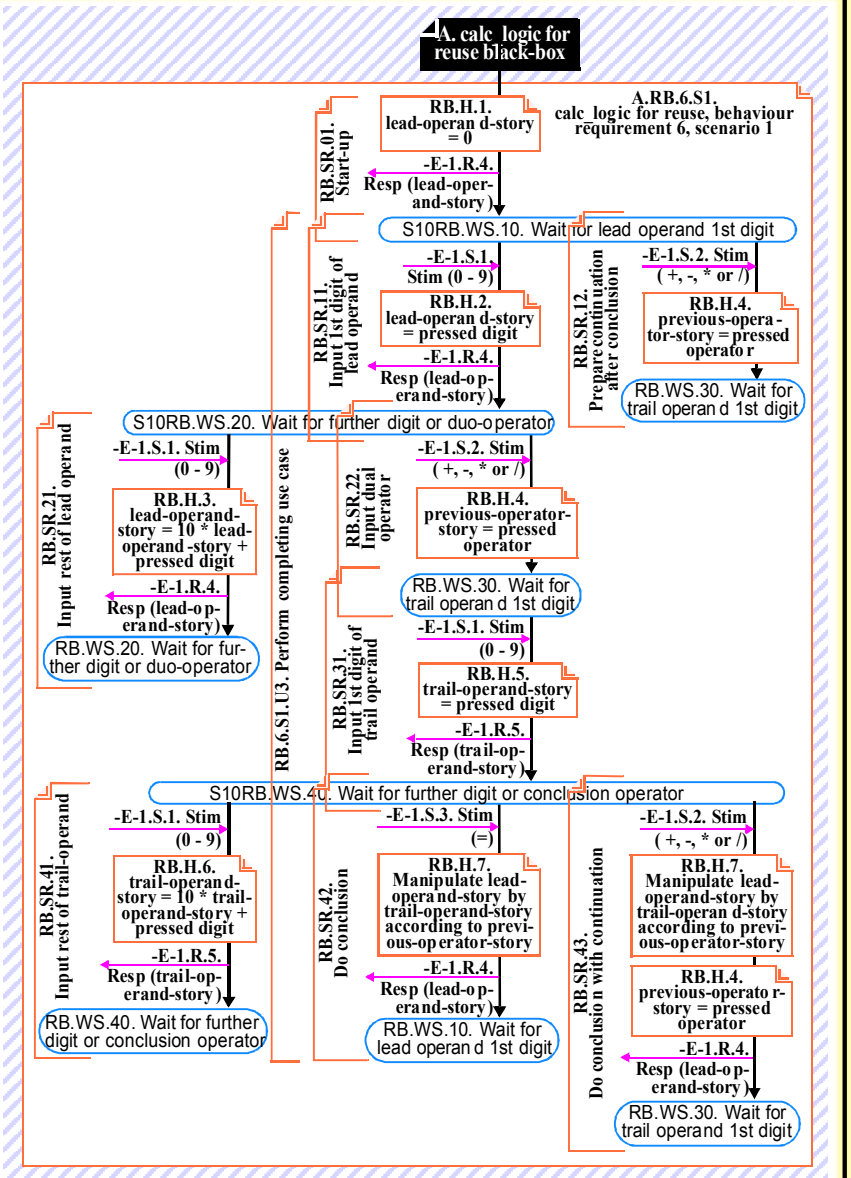
calc_logic for reuse behaviour requirements

Black-box behaviour wait-states, SR-unities and histories

A. RB. calc_logic for reuse behaviour requirement	
A. RB.WS.10.	**Wait for lead operand 1st digit**
A. RB.WS.20.	**Wait for further digit or duo-operator**
A. RB.WS.30.	**Wait for trail operand 1st digit**
A. RB.WS.40.	**Wait for further digit or conclusion operator**
A. RB.SR.01.	**Start-up**
A. RB.SR.11.	**Input 1st digit of lead operand**
A. RB.SR.12.	**Prepare continuation after conclusion**
A. RB.SR.21.	**Input rest of lead operand**
A. RB.SR.22.	**Input dual operator**
A. RB.SR.31.	**Input 1st digit of trail operand**
A. RB.SR.41.	**Input rest of trail-operand**
A. RB.SR.42.	**Do conclusion**
A. RB.SR.43.	**Do conclusion with continuation**
A. RB.H.1.	**lead-operand-story = 0**
A. RB.H.2.	**lead-operand-story = pressed digit**
A. RB.H.3.	**lead-operand-story = 10 * lead-operand-story + pressed digit**
A. RB.H.4.	**previous-operator-story = pressed operator**
A. RB.H.5.	**trail-operand-story = pressed digit**
A. RB.H.6.	**trail-operand-story = 10 * trail-operand-story + pressed digit**
A. RB.H.7.	**Manipulate lead-operand-story by trail-operand-story according to previous-operator-story**

(continued) EXAMPLE calc_logic: Graphical use case scenario

- Component containing the intelligence in a calculator
- Built for reuse in
  - Standalone calculator
  - Windows calculator application



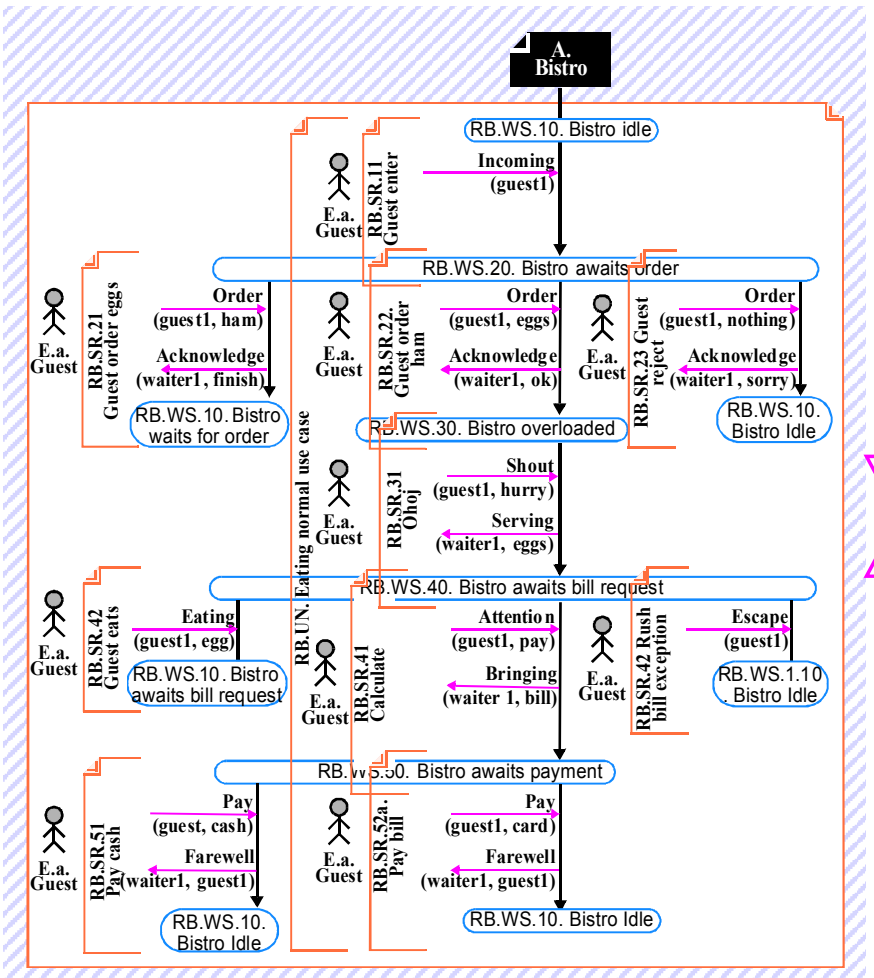
EXAMPLE calc-_logic: Test use case

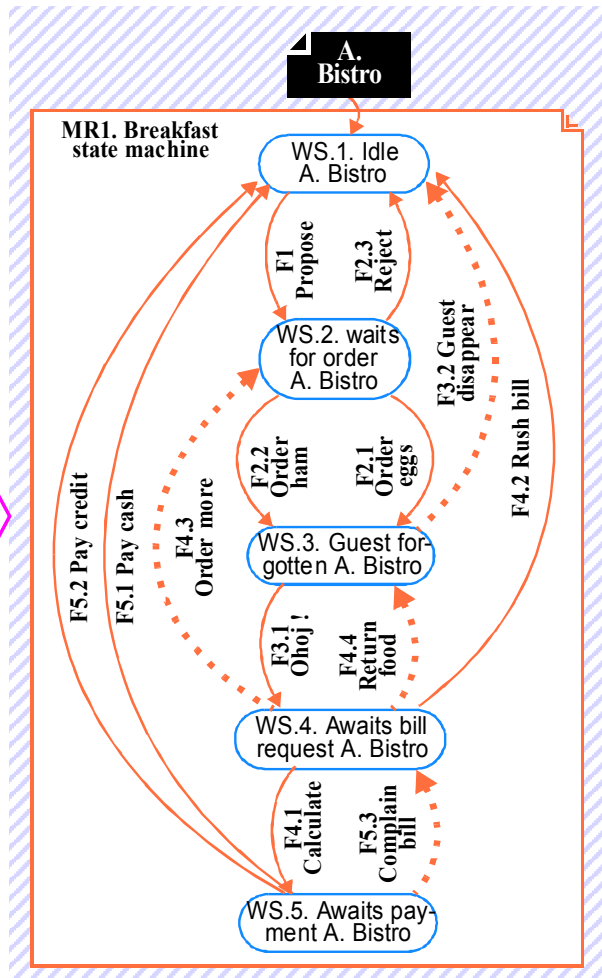
- Use case can be the form to hold a test case
- Execute test use cases in the mind to check requirements

Use case	SR-Unity	Stimulus	Response	lead-operand-story	previous-operator-story	trail-operand-story
	RB.SR.01. Start-up	**Start of A.**	**E-1.R.4. Show (0)**	RB.H.1. = 0		
RB.WS.10. Wait for lead operand 1st digit						
RB.6.S1.U4. Perform repeating use case	**RB.SR.11. Input 1st digit of lead operand**	**E-1.S.1 Press (2)**	**E-1.R.4. Show (2)**	RB.H.2. = 2		
	RB.WS.20. Wait for further digit or duo-operator					
	RB.SR.21. Input rest of lead operand	**E-1.S.1 Press (4)**	**E-1.R.4. Show (24)**	RB.H.3. = 10 * 2 + 4		
	RB.WS.20. Wait for further digit or duo-operator					
	RB.SR.22. Input dual operator	**E-1.S.2. Press (-)**		24	RB.H.4. = -	
	RB.WS.30. Wait for trail operand 1st digit					
	RB.SR.31. Input 1st digit of trail operand	**E-1.S.1 Press (6)**	**E-1.R.5. Show (6)**	24	-	RB.H.5. = 6
	RB.WS.40. Wait for further digit or conclusion operator					
	RB.SR.43. Do conclu-sion with continuation	**E-1.S.2. Press (+)**	**E-1.R.4. Show (18)**	RB.H.7. = 24 - 6 = 18	RB.H.4. = +	6
RB.WS.30. Wait for trail operand 1st digit						
RB.6.S1.U3. Perform completing use case	**RB.SR.31. Input 1st digit of trail operand**	**E-1.S.1. Press (4)**	**E-1.R.5. Show (4)**	18	+	RB.H.5. = 4
	RB.WS.40. Wait for further digit or conclusion operator					
	RB.SR.42. Do conclu-sion	**E-1.S.3. Press (=)**	**E-1.R.4. Show (22)**	RB.H.7. = 18 + 4	+	4
RB.WS.10. Wait for lead operand 1st digit						
RB.6.S1.U5. Perform repeating after conclusion use case	**RB.SR.12. Prepare continuation after con-clusion**	**E-1.S.2. Press (/)**		22	RB.H.4. = /	4
	RB.WS.30. Wait for trail operand 1st digit					
	RB.SR.31. Input 1st digit of trail operand	**E-1.S.1. Press (2)**	**E-1.R.5. Show (2)**	22	/	RB.H.5. = 2
RB.WS.40. Wait for further digit or conclusion operator						
	RB.SR.42. Do conclu-sion	**E-1.S.3. Press (=)**	**E-1.R.4. Show (11)**	RB.H.7. = 22 / 2	/	2
RB.WS.10. Wait for lead operand 1st digit						

EXAMPLE Bistro: State machine

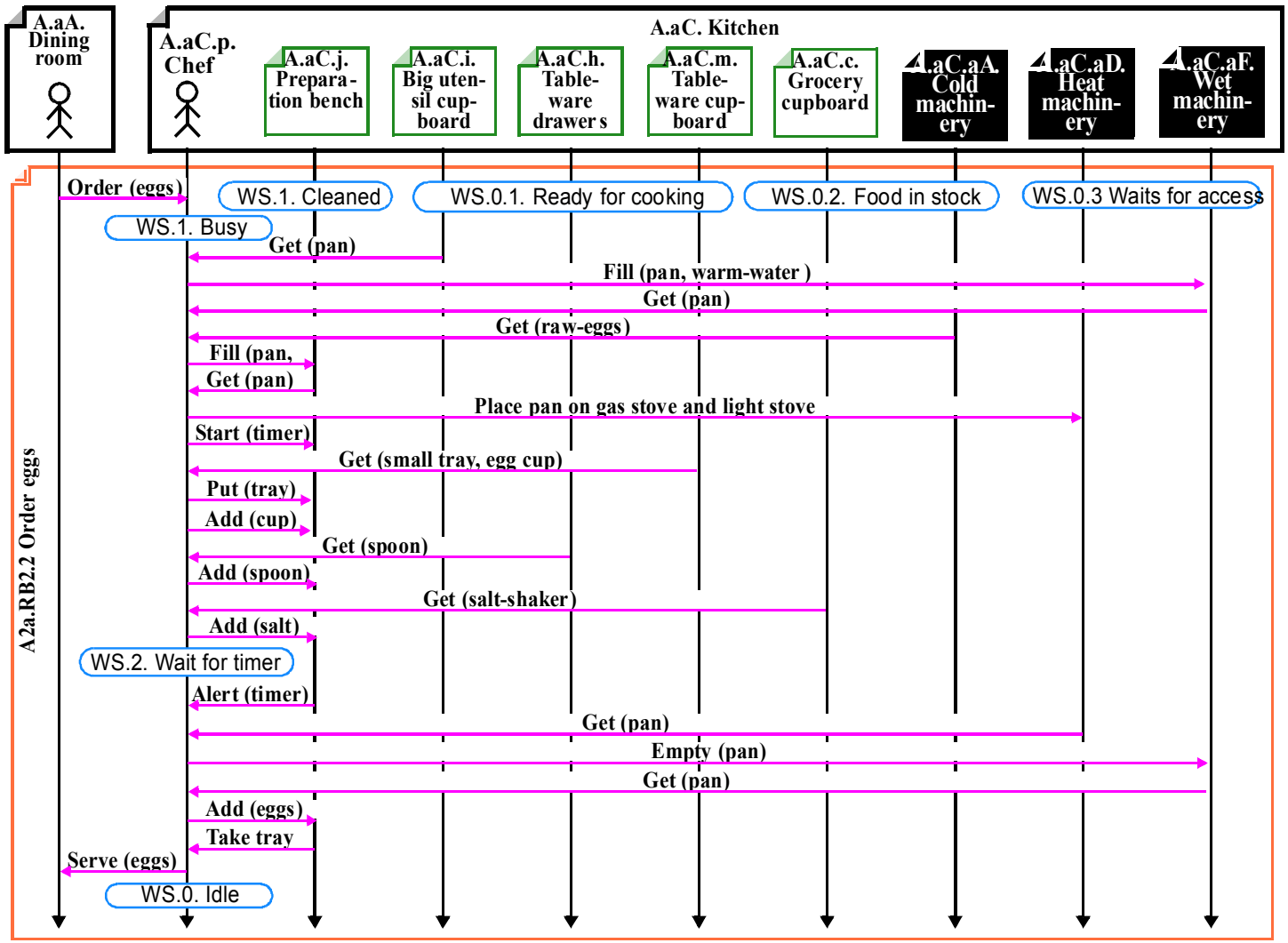
- Short-form for use case scenario
- Contain no
  - Stimuli
  - Responses
  - Multiple treads
- Identifies exception transitions
  - Dashed in figure





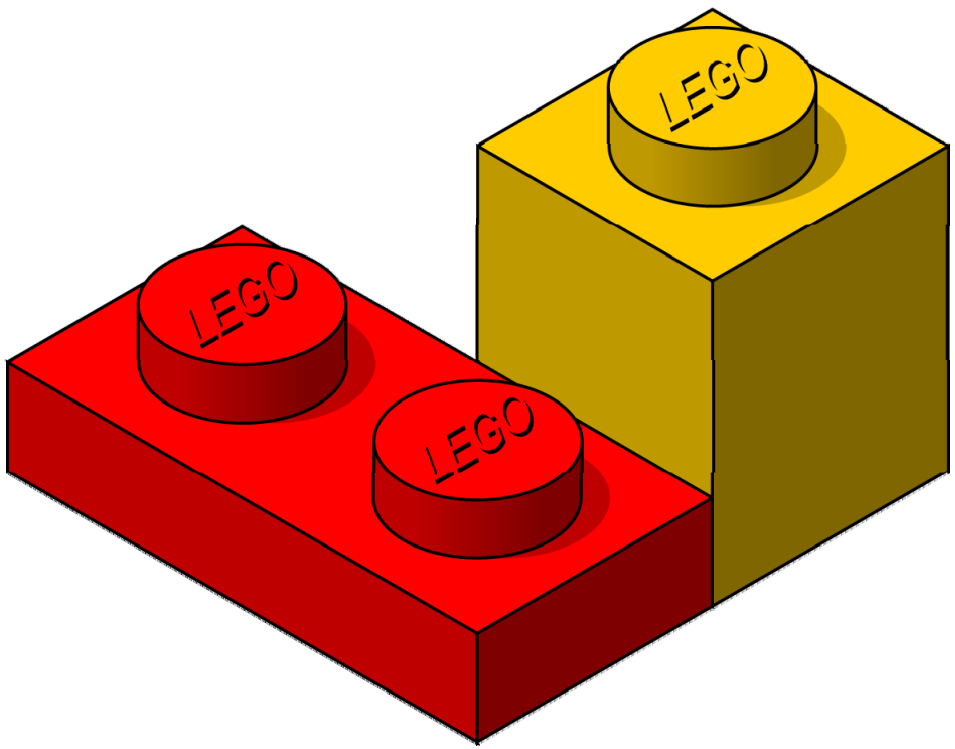
EXAMPLE Bistro: Message sequence diagram

- Abbreviated MSC
- Use case penetrating architecture
- Each element can have wait-states



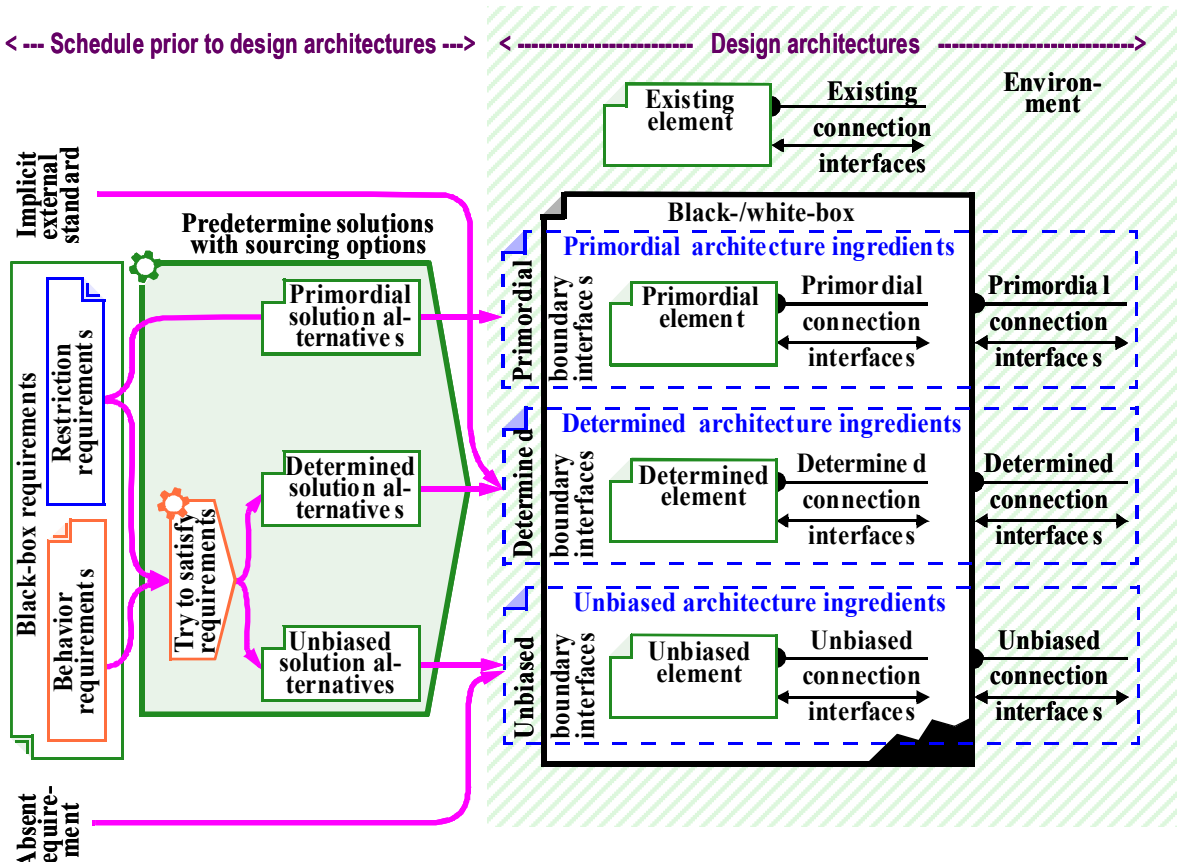
Predetermine solutions and suppliers

Solution alternatives off the shelf?

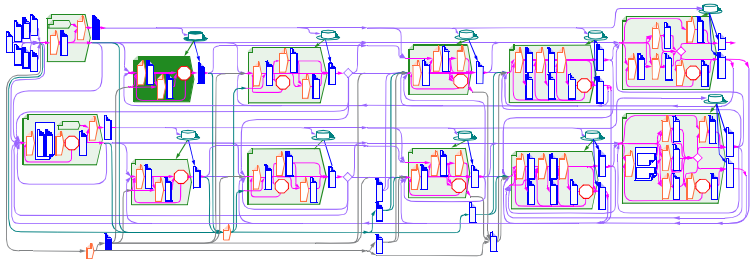


Trying out solution alternatives from requirements

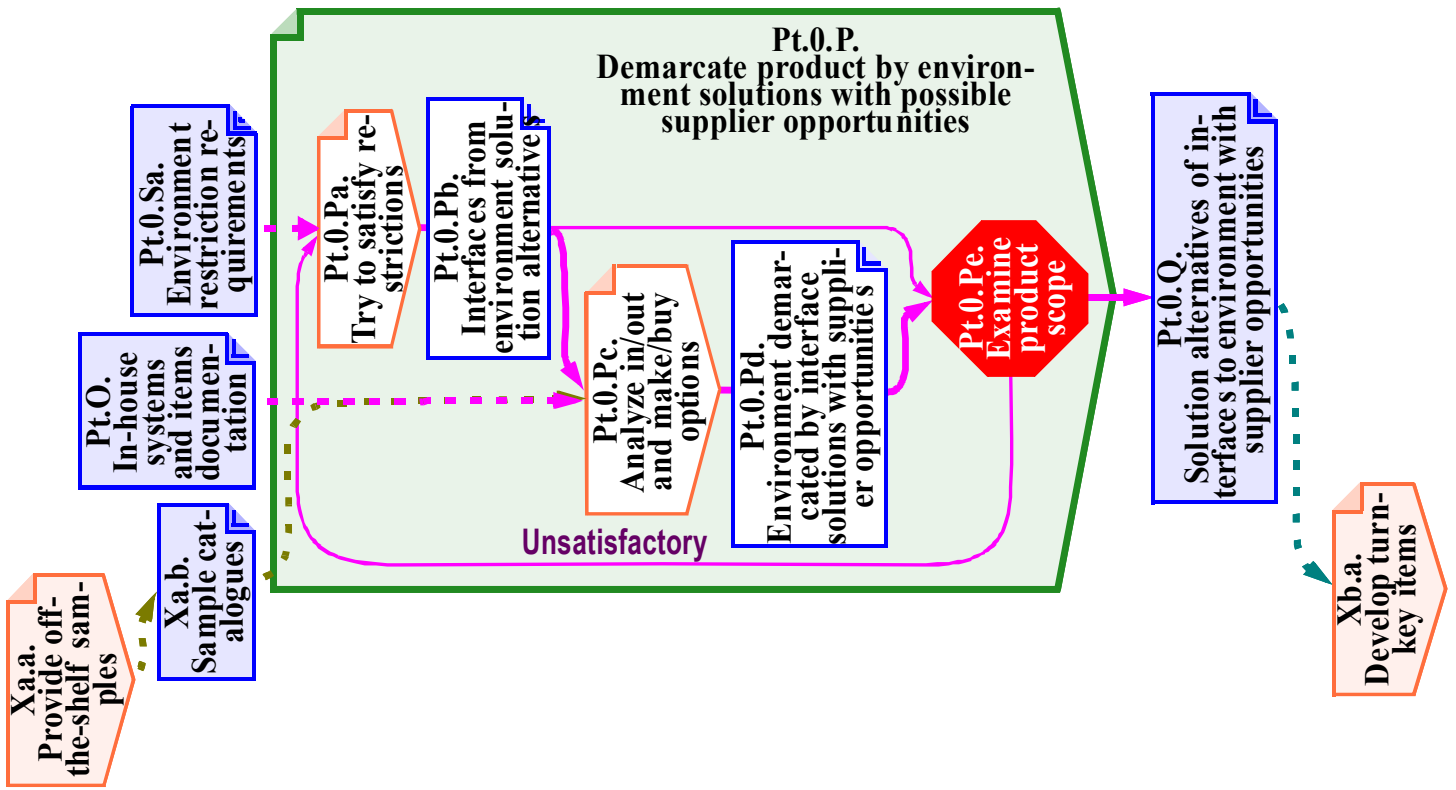
- Primordial design often results from restriction requirements
  - stakeholders decide design
- Unbiased design often results from behavior requirements or absence of requirements
  - technicians decide design
- Determined design are tried out to best fit requirements and standards
  - both stakeholders and technicians decide design



Pt.0.P. Demarcate product by environment solutions with possible supplier opportunities

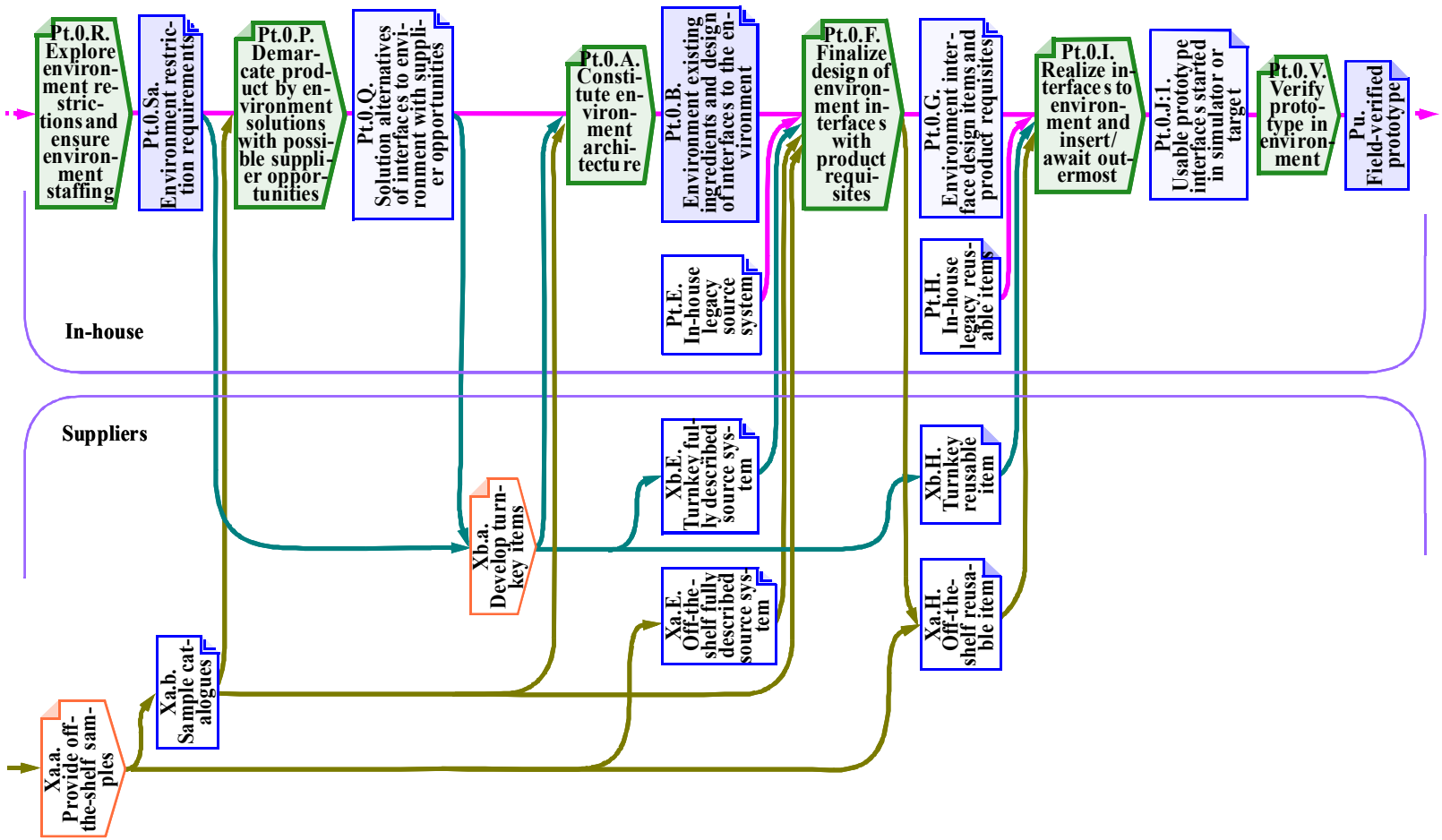


- Try satisfying restrictions by solution alternatives
- Based on solution alternatives, decide what is inside and outside of this product
- Propose suppliers and samples



Sourcing interfaces to environment from suppliers

- Suppliers must be usable along the whole development



EXAMPLE House: Environment solution alternatives

- For each restriction requirement, document as many reasonable imaginable solutions as possible
- For each solution, decide if to be included in product to be developed
- If beneficial for later design, assess each solution alternative

Environment solution alternatives	Cost assessment	Environment restriction
E.SL1.a. Keep distance to shoreline as close as possible to 150 m **E.SL1.b. Build on higher land to keep lake view** **E.SL1.c. Build higher house to keep lake view** • Building plot	**Exclude these solutions**	E.R.1. Swedish environmental code 7th chapter
E.SL2.a. Obtain extra 2 000 m² land, to ensure a building area of 100 m² **E.SL2.b. Obtain no extra land and build partly under ground** **E.SL2.c. Obtain no extra land and build smaller** • Building size	**Exclude these solutions**	E.R.2. Malmö urban planning and local construction ordinance
E.SL3. When interfaces to the environment is constructed, existing neighbour environment may not be damaged.	**Exclude these solutions**	R.3. Swedish national building regulations
E.SL4. Ramp from driveway to entry for disabled persons	**~ ? 1 800**	
E.SL5. The building plot must be located to an attractive area.	**Exclude these solutions**	R.4. Swedish Financial Supervisory Authority
E.SL6.a. Ø 32 mm gas pipe from street • Gas is available in street **E.SL6.b. 3 m³ or 7 m³ rental tank close to house** • Gas tank for free, included in gas deliveries	**~ ? 7 000** **~ ? 500**	E.R.5. All communal connections available shall be used to make the house functional and comfortable.
E.SL7.a. 16 Amperes from overhead lines **E.SL7.b. 32 Amperes from underground cable** • Electricity	**~ ? 1 000** **~ ? 3 000**	
E.SL8.a. Drilled well with pump and purifier **E.SL8.b. Ø 32 mm water pipe from street** **E.SL8.c. Pipe from lake with pump and purifier** • Water	**~ ? 11 000** **~ ? 4 000** **~ ? 8 000**	
E.SL9.a. Waste water treatment to lake recipient **E.SL9.b. Ø 125 mm pipe to street waste water** • Sewage	**~ ? 6 300** **~ ? 2 000**	
E.SL10.a. Basalt covered 2,4 m wide road **E.SL10.b. 4 basalt covered parking places, 2 m road extension at street, with 0,5 m wide basalt covered footpath to house** • Driveway	**~ ?15 000** **~ ? 2 000**	
Material min cost sum **Material max cost sum**	**~ ? 11 300** **~ ? 44 100**	

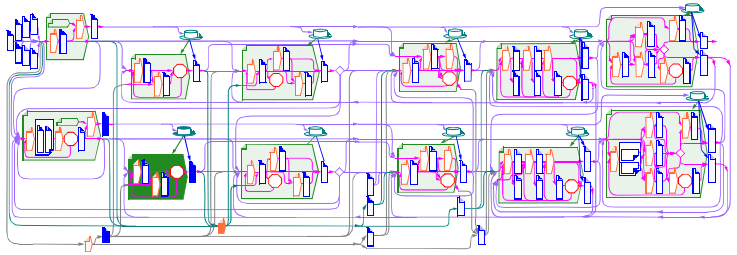
EXAMPLE House environment: Excluded and included solution alternatives with supplier opportunities

- Demarcate the product from the existing environment to prevent the product from swelling out
- Voluminous solutions for interfaces to environment are excluded from development
- Definitely needed interfaces to environment are included

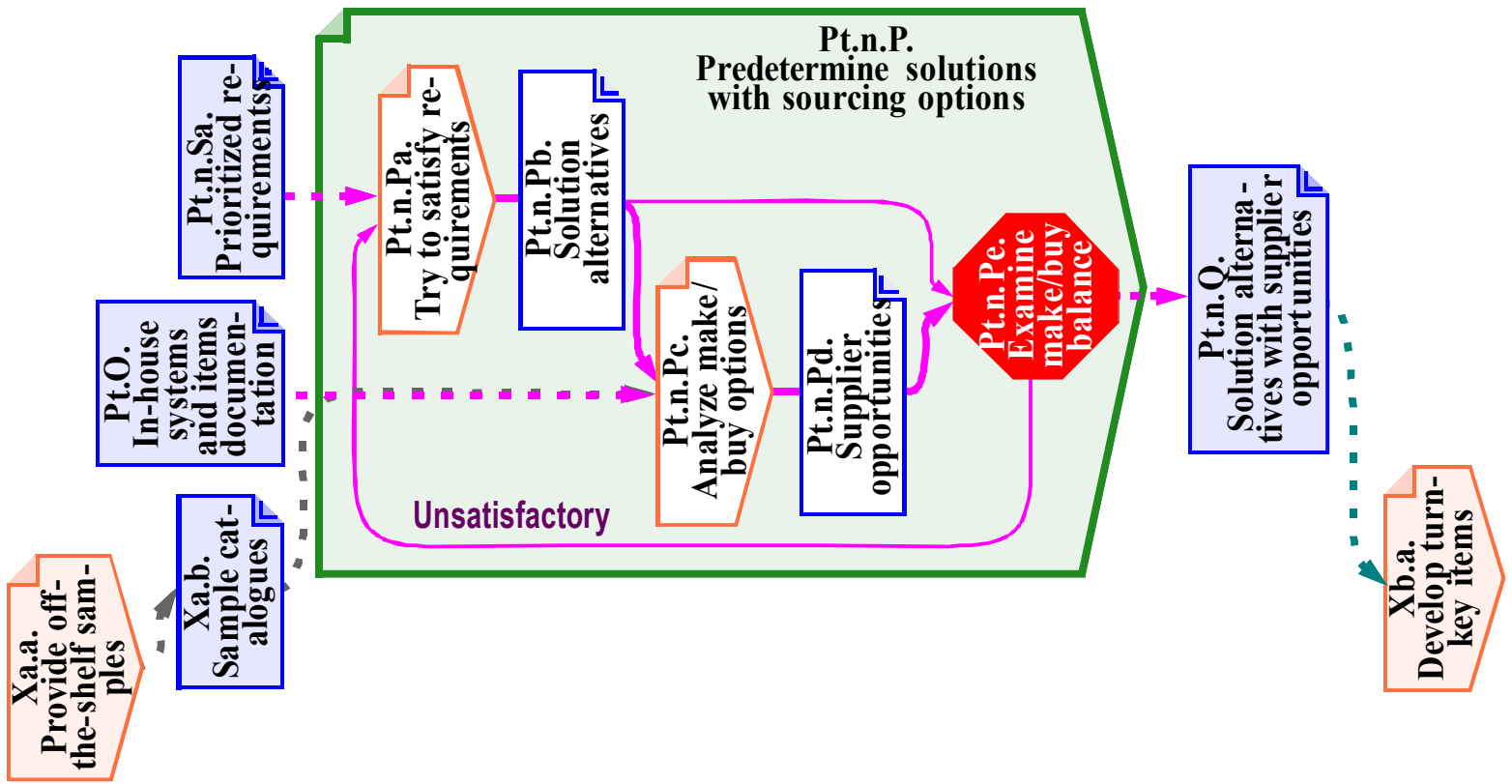
Excluded environment solutions	Responsible	Environment solution alternatives
SX.1 Buy beachside building land larger than 2 000 m² with possible house ground at 150 m from the lake	**House proprietor**	Building plot E.SL1.a., E.SL1.b., E.SL1.c. E.SL2.a., E.SL2.b., E.SL2.c.
SX.2. Damages on neighbours environment	**House proprietor**	E.SL3.
SX.3. Lake view is more than good enough	**House proprietor**	E.SL5.
SX.4. Water and electricity supply from environment during building the house	**House proprietor**	Preliminary electricity and water E.SL7.a., E.SL7.b., E.SL8.a., E.SL8.b., E.SL8.c.
SX.5. Connection fees to authorities and commune	**House proprietor**	Interface connections E.SL7.a., E.SL7.b., E.SL8.a., E.SL8.b., E.SL8.c., E.SL9.a., E.SL9.b., A.SL7.

Included environment solution with supplier opportunities	Environment solution alternatives
SU.10. Builder's suppliers • Use ordinary building material, from selected builder suppliers.	Electricity, water and sewage E.SL7.a., E.SL7.b., E.SL8.a., E.SL8.b., E.SL8.c.
SU.11. Gas supplier • Use only authority approved equipment, Use tank supplied by gas provider.	Gas pipe and tank E.SL6.a., E.SL6.b.
SU.12. Ext Bas AB • Use stone sample H34	Driveway and ramp E.SL4., E.SL10.a.
SU.13. Local supplier • Use natural material that harmony with the environment from local suppliers	Driveway E.SL9.a., E.SL9.b.

Pt.n.P. Predetermine solutions and sourcing options



- Try to satisfy requirements when designing black-box
- Propose samples and suppliers



EXAMPLE calc_logic: Solution alternatives and supplier opportunities

calc_logic for reuse supplier opportunities	calc_logic for reuse solution alternative
SU.12. Inhouse project DG3	A. calc_logic for reuse A.SL1., A.SL2., A.SL3., A.SL5., A.SL6., A.SL9., A.SL10., A.SL11., A.SL12.
SU.13. Well direct	Development system

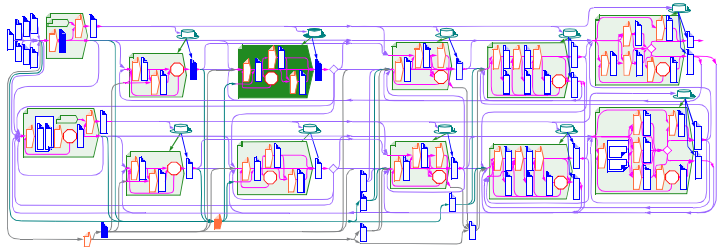
- Try finding solution alternatives for outmost black-box requirements
- In this example alternatives are few
- Most development will be made inhouse
  - Typical for programs

calc_logic for reuse solution alternatives	Assessment	calc_logic for reuse requirement	Priority
A.SL1. calc_logic must internally handle 64 buttons.	No extra cost	A.RR.1.1. The calc_logic element must be able to process 16 button stimuli at minimum and be future-proof to process many more button stimuli.	
A.SL2. Up to 128 digits must be internally handled, even if not all are accurate. A.SL3. Handling must include both numeric and non-numeric characters	No extra cost	A.RR.2.1. The calc_logic element must be able to process 8 digits responses, and must be future-proof to a much large amount of digits.	
A.SL4. Any mathematical coprocessor shall never be used by calc_logic, but instead the compiler with built-in library or external library.	Saving cost	A.RR.3.1. To reduce complexity from unpredictable existence of coprocessor, a math-copro cessors shall not be used at all.	
A.SL5. Develop first calc_logic on a host development system supporting ~ 15 digits precision. A.SL6. Then port calc_logic to a Windows hosted development system for micro controllers supporting 7 digits floating point precision A.SL7. Then port calc_logic to a micro controller experimental board A.SL8. Then port calc_logic to the target standalone pocket calculator A.SL9. When available, procure compiler for the windows supporting better double floating point precision A.SL10. If different precision or different target demands calc_logic specific solutions, these must be hidden behind compiler switches.	~ 40 000 ~ 1 000 ~ 2 000 ~ 2 000 No extra cost Small cost	A.RR.4.1. It is good enough with a maximum calculation precision of ~ 15 digits during host development of the calc_logic program.	
A.SL11. For development of calc_logic in host, development boards and target, use development systems having open and documented mechanisms for function call and return.		A.RR.5.1. The calc_logic element must in first version be portable to the windows calculator application and to the pocket standalone calculator.	
A.SL12. Make implementation in C, and use scenario wait-states in program.	Cost ~ 20 000 (excluded compiler and math library)	A.RB.6.S1. calc_logic for reuse, behaviour requirement 6, scenario 1	

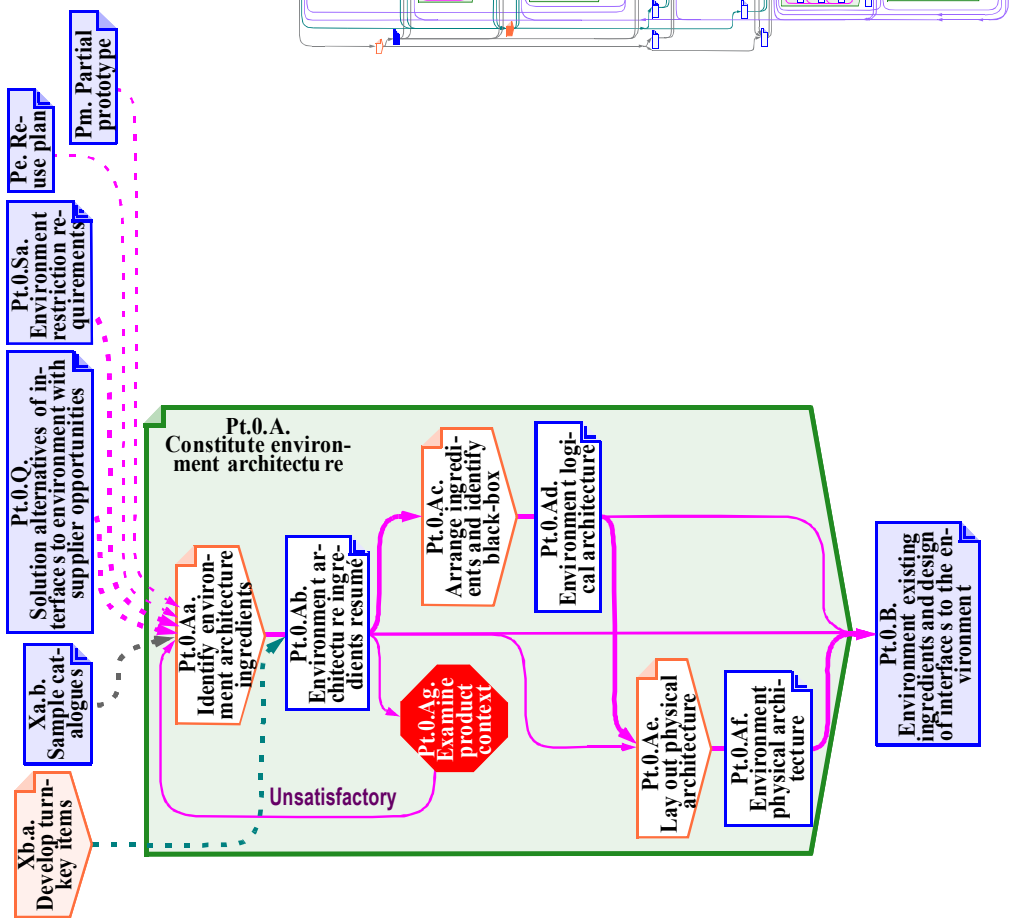
Design architectures

Time and money for architects?

Pt.0.A. Constitute environment architecture



- Interfaces to environment shall now be designed
- Alternative solution are selected and decided, including eventual supplier samples, and noted in architecture resumé
- Ingredients in the resumé are arranged in an environment logical architecture
- Ingredients are depicted in an environment physical architecture

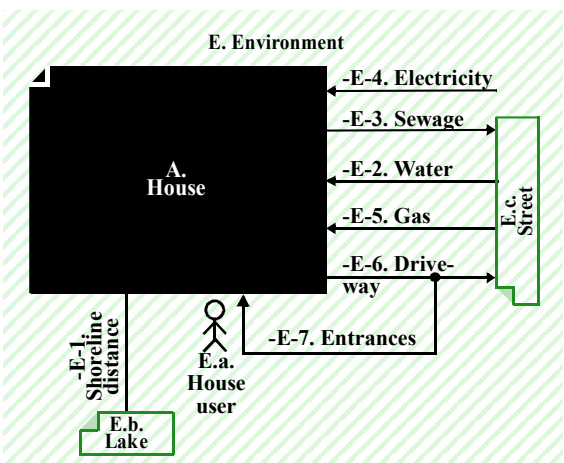


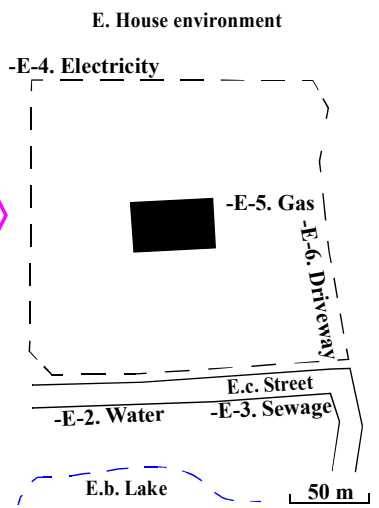
EXAMPLE House: Environment architectures

House environment architecture resumé

Environment ingredients		Existing environment and solution alternatives to interface it
E.	House environment	The existing surrounding to the house E.SL3. When interfaces to the environment is constructed, existing neighbour environment may not be damaged.
E.a.	House user	Proprietor, relatives and guests coming from the environment
E.b.	Lake	Existing beautiful lake, south of the planned house
E.c.	Street	Existing residential street, east of the planned house
-E-1.	Shoreline distance	Product interface to be developed E.SL1.a. Keep distance to shoreline as close as possible to 150 m
-E-2.	Water	Product interface to be developed E.SL8.b. Ø 32 mm water pipe from street
-E-3.	Sewage	Product interface to be developed E.SL9.b. Ø 125 mm pipe to street waste water
-E-4.	Electricity	Product interface to be developed E.SL7.b. 32 Amperes from underground cable
-E-5.	Gas	Product interface to be developed E.SL6.b. 3 m³ or 7 m³ rental tank close to house Must go shortest way from outer wall to inside burner.
-E-6.	Driveway	Product interface to be developed E.SL10.a. Basalt covered 2,4 m wide road. Stone sample H34 from Ext Bas AB
-E-7.	Entrances	Product interface to be developed E.SL4. Ramp from driveway to entry for disabled persons
A.	House	Product to be developed E.SL2.a. Obtain extra 2 000 m² land, to ensure a building area of 100 m². E.SL5. The building plot must be located to an attractive area.

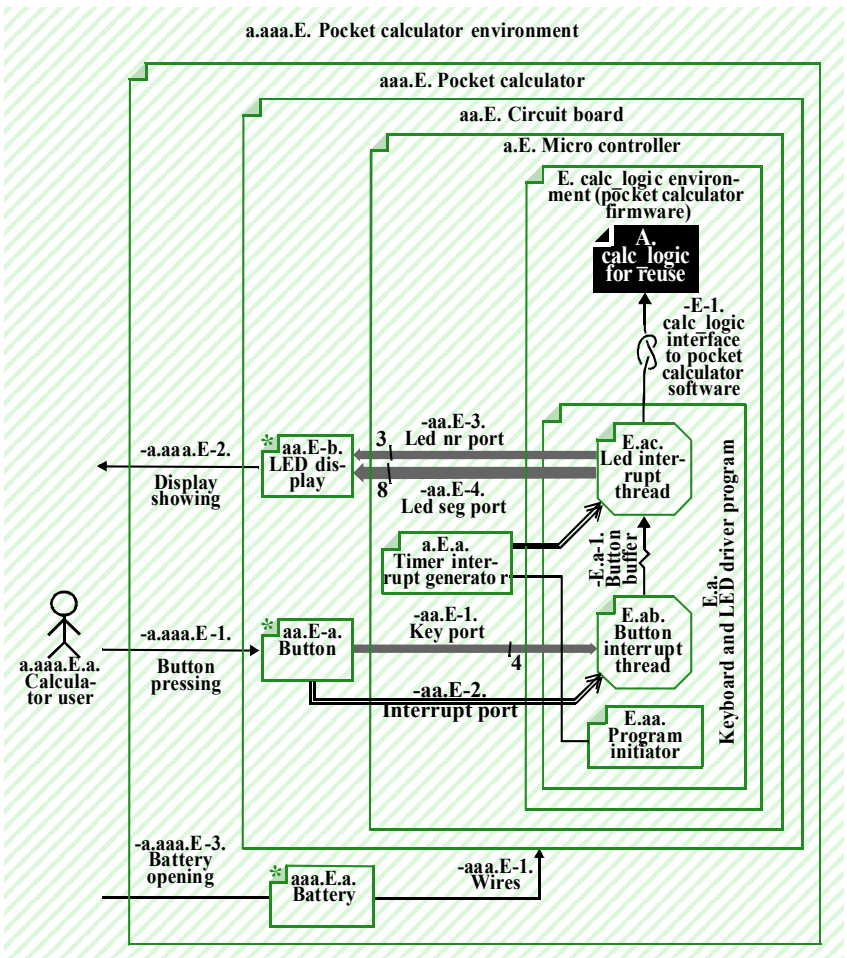
- Environment architectures
  - Resumé
  - Logical
  - Physical

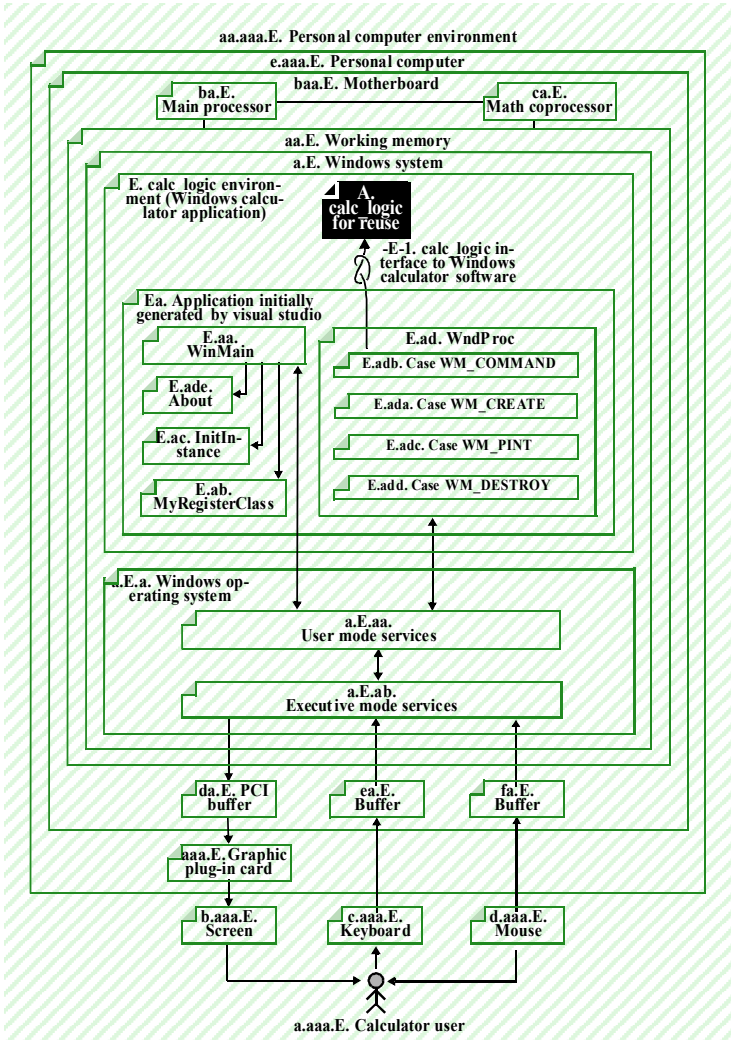




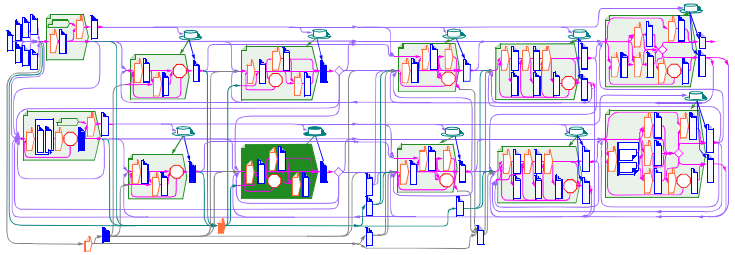
EXAMPLE Calc_logic: Environment logical architecture

- Same reusable component with very different environments
- One architecture for the Windows application and one for the pocket calculator

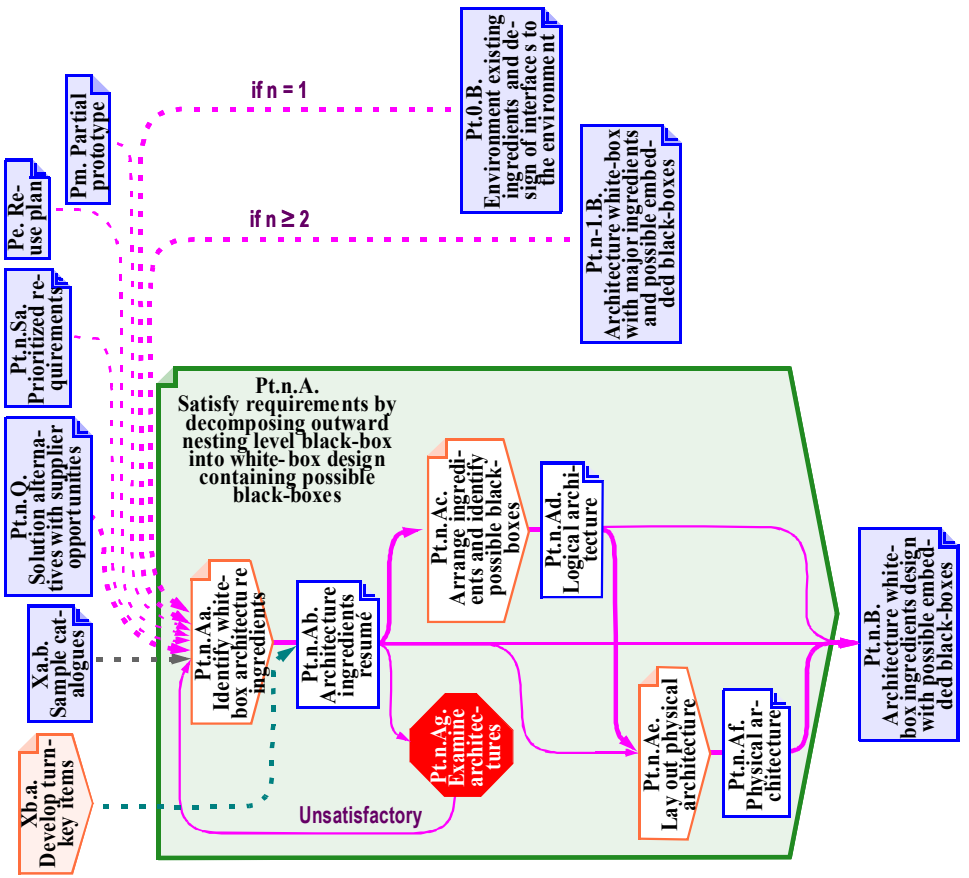




Pt.n.A. Satisfy requirements by decomposing outward nesting level black-box into white-box design containing possible black-boxes

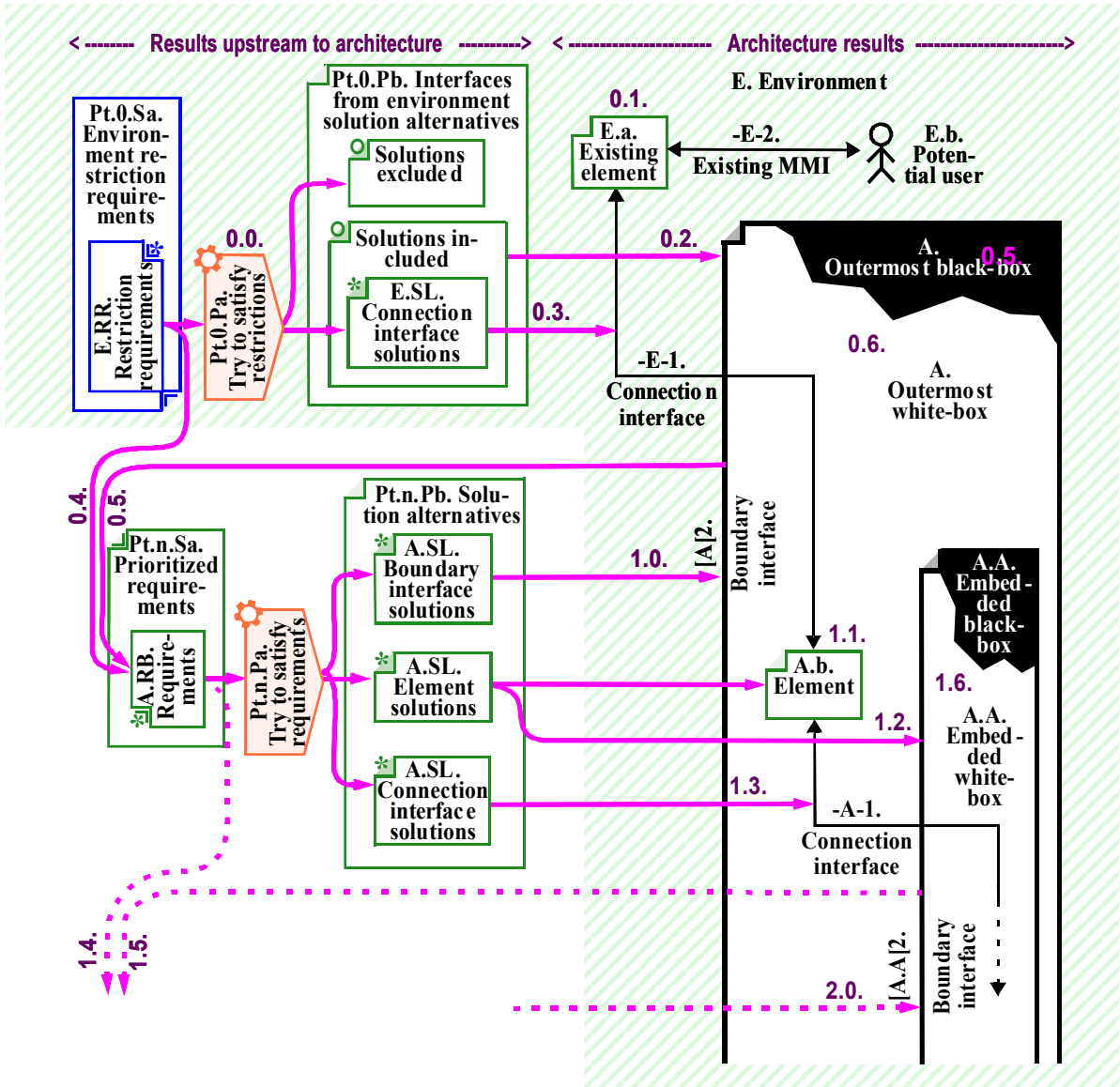


- Black-boxes from outward level shall now be opened and designed
- Alternative solution are selected and decided, including supplier samples, and noted in the architecture resumé
- Ingredients in the resumé are arranged in a physical architecture
- Ingredients are depicted in a physical architecture



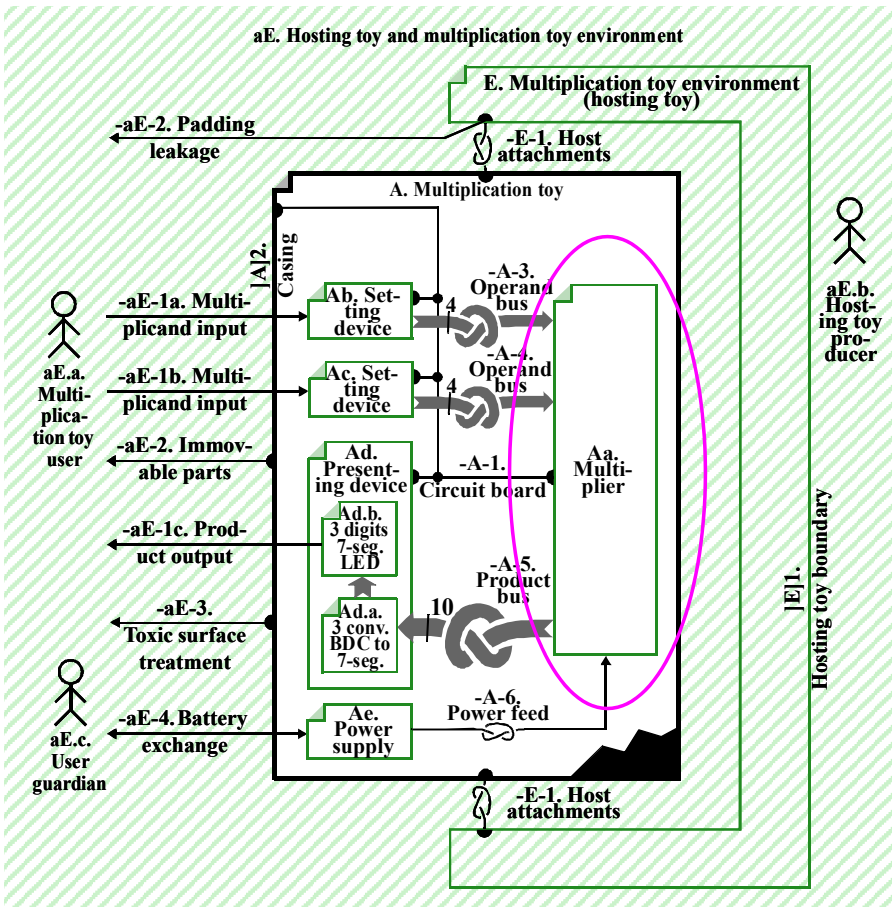
Connection and boundary interfaces

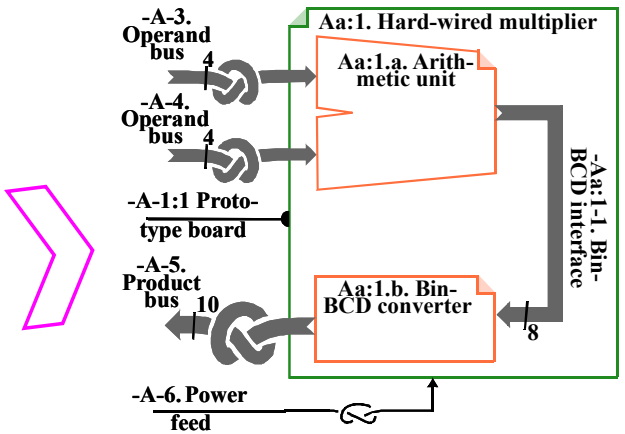
- Always development outside-in
- Connection interfaces are keeping elements in contact
  - Belongs to the outermost element where its ends appear
  - designated like –E–1.
- Boundary interfaces are keeping elements apart
  - Belongs to the element it surrounds
  - designated like [A[2.

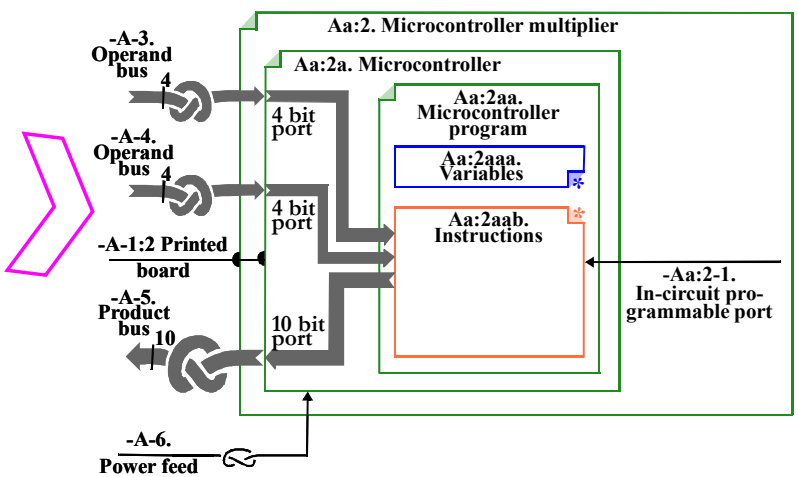


EXAMPLE Multiplication Toy: Logic architecture with exchangeable multiplier

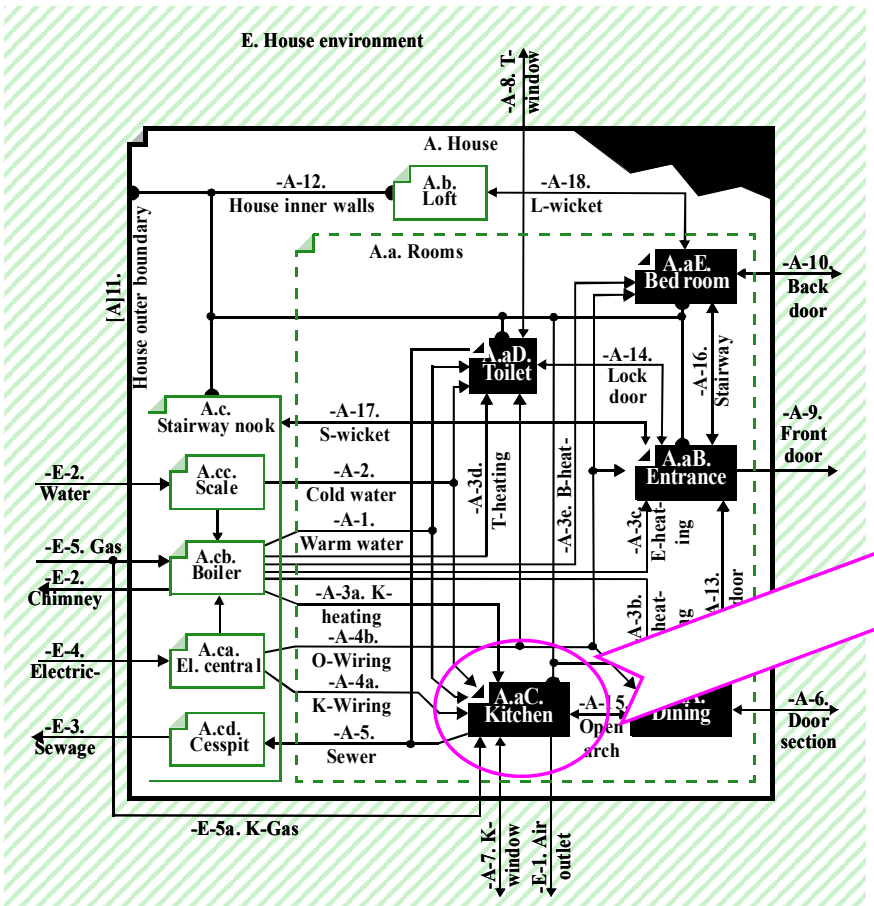
- Multiplier realized with two different technologies
  - Hardwired logical gates
  - Micro-controller with program

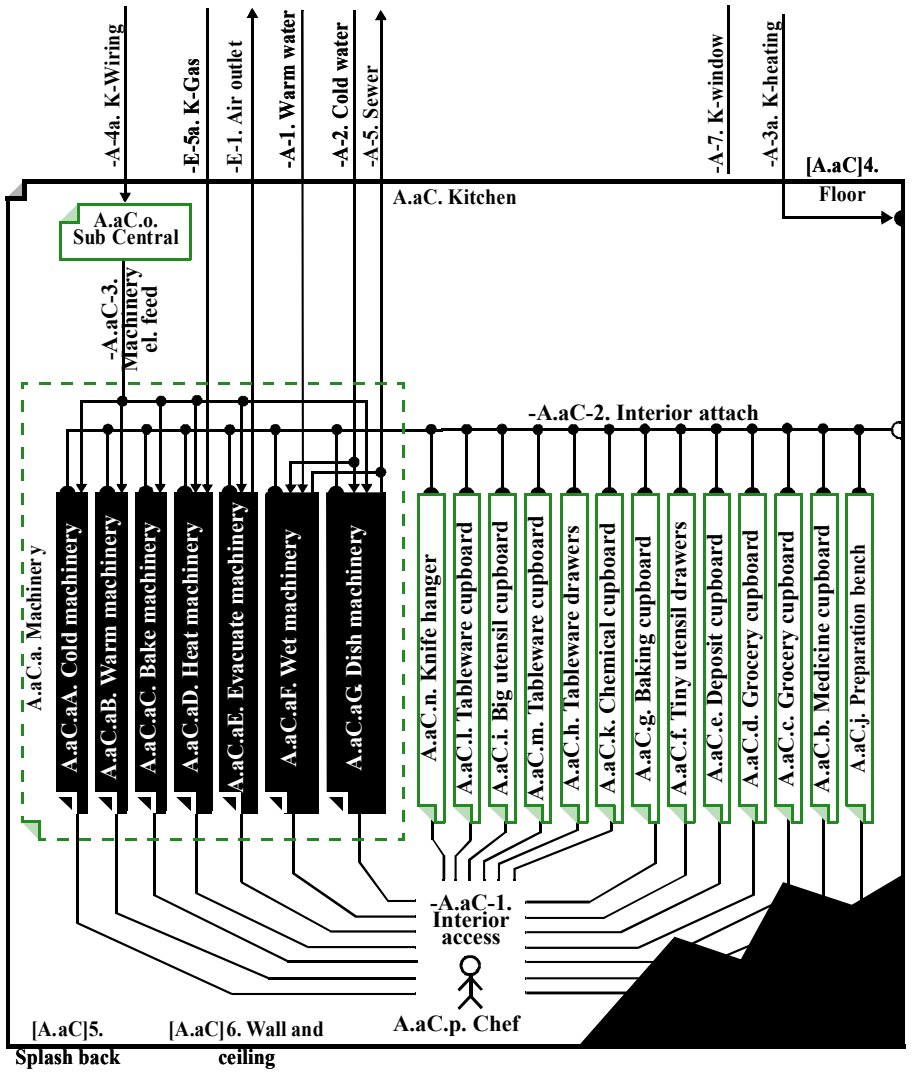






EXAMPLE House: Black-box in house decomposed to kitchen white-box

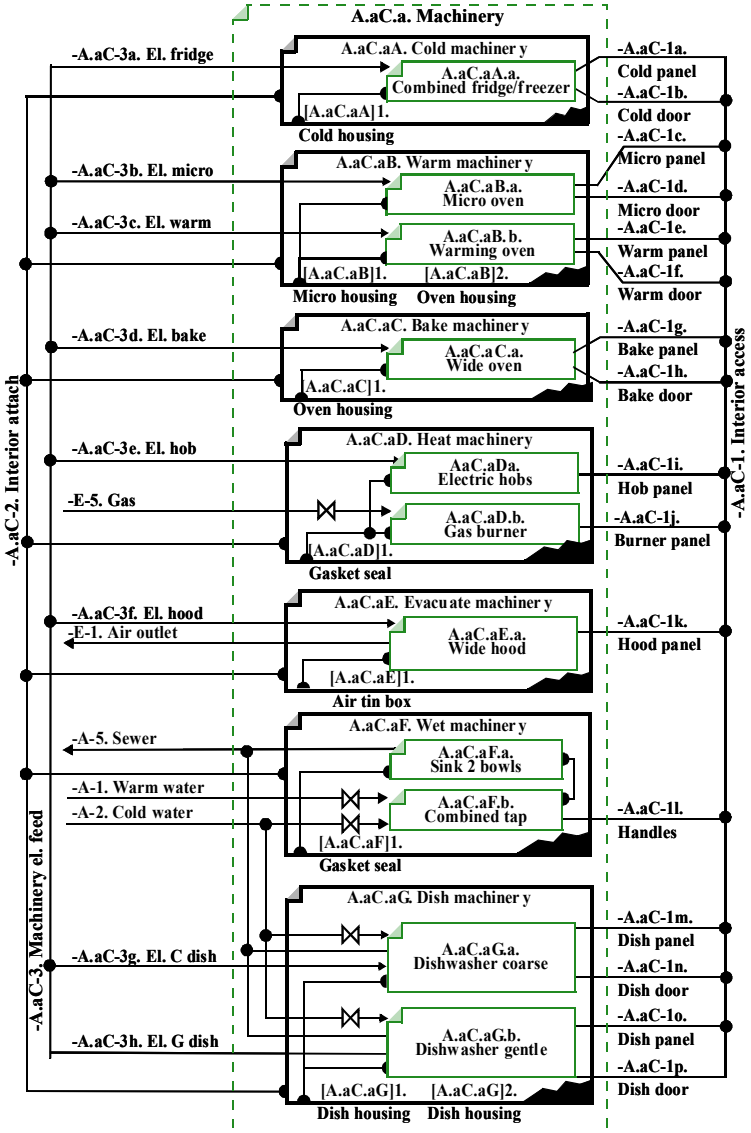




EXAMPLE House: Machinery logical and physical architecture

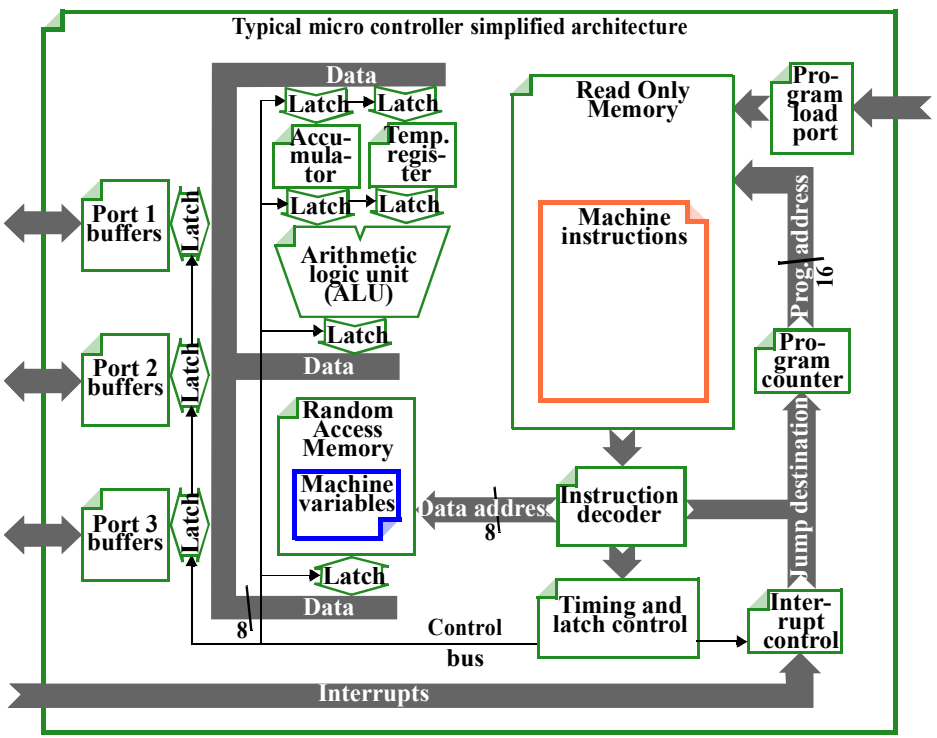
- Physical architecture

- Logical architecture

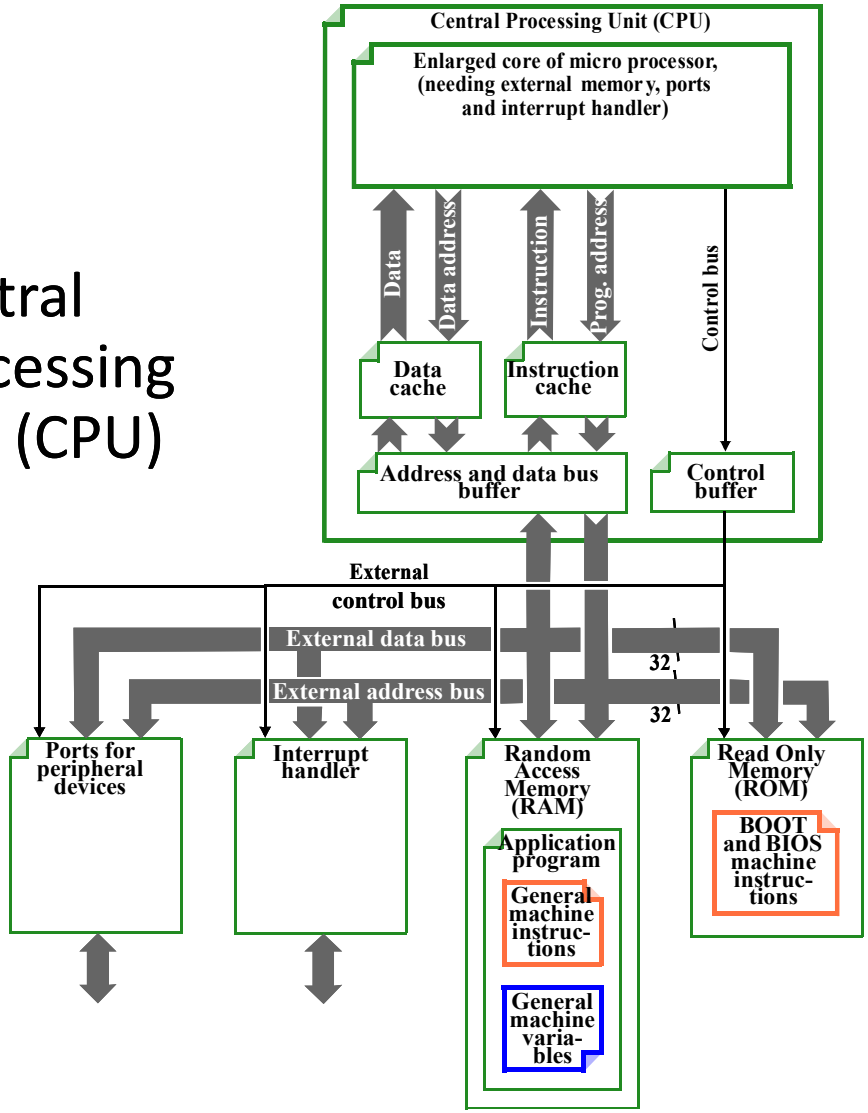


Micro controller versus central processing unit (CPU)

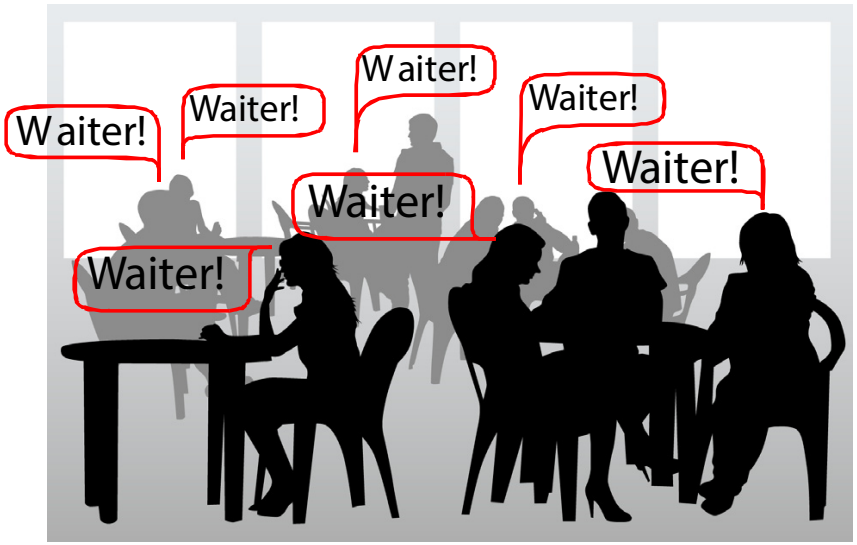
- Micro controller



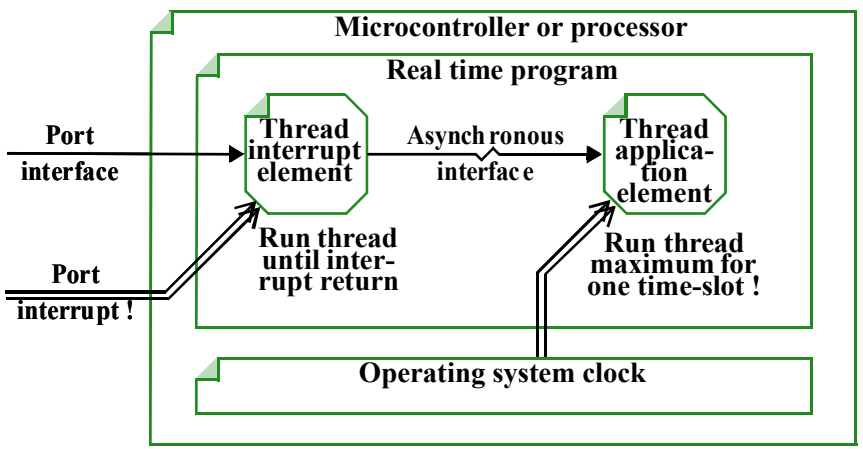
- Central processing unit (CPU)



Interrupt and threads

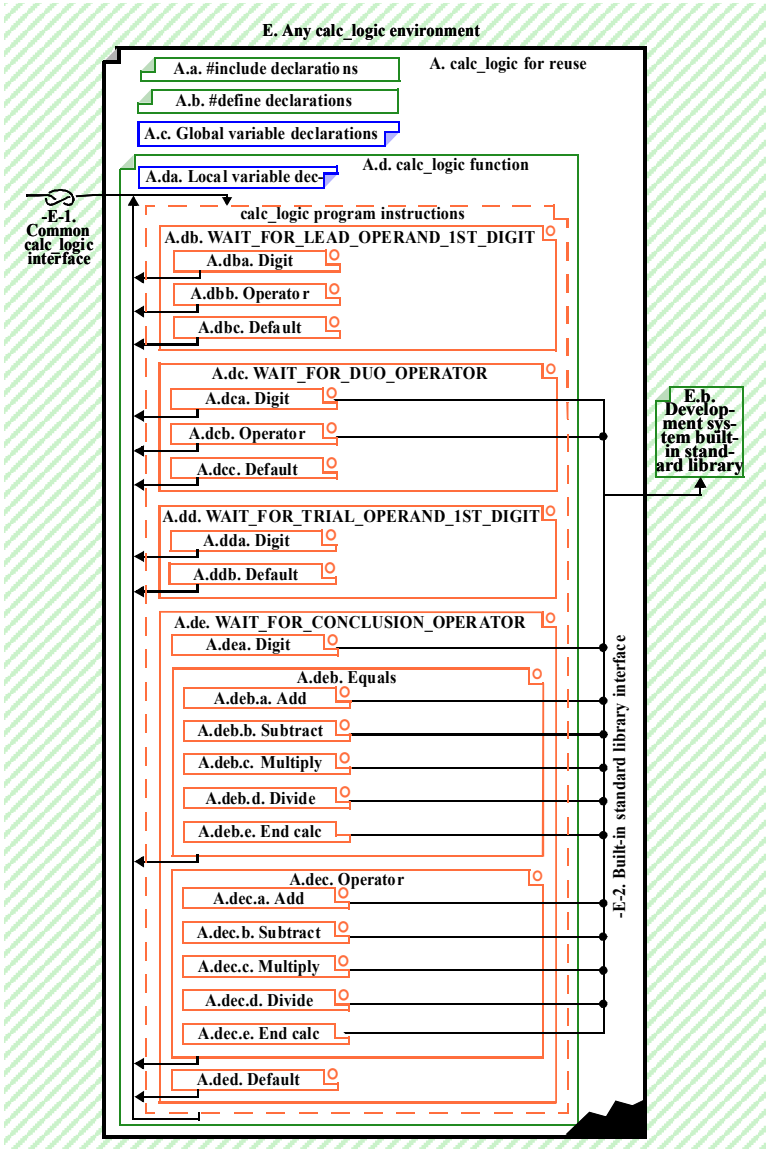


- High level thread constructions take care of unsynchronized interrupt stimuli
  - To avoid that all program paths must check all imaginable stimuli sources
- Organize one thread to take care of each source of scenario stimuli
  - Preferably regard each tread as black-box
- Let each tread be instant to response
- When time, let threads synchronize and finalize what not instantly was done

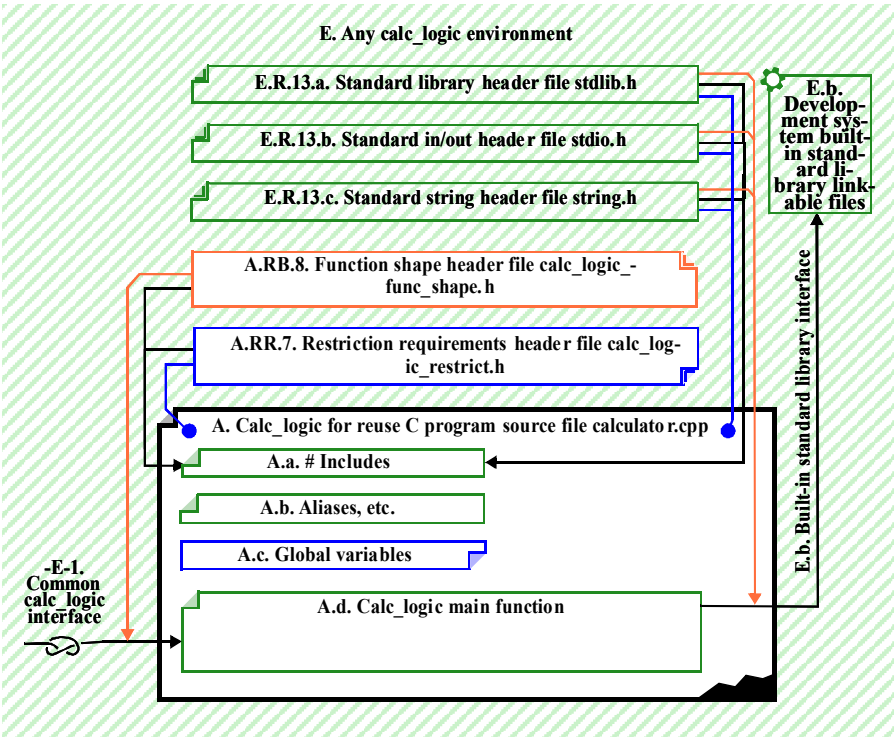


EXAMPLE calc_logic: Logical and physical program architecture

- Logical architecture presumes high level program
- Architecture may imitate specification scenario

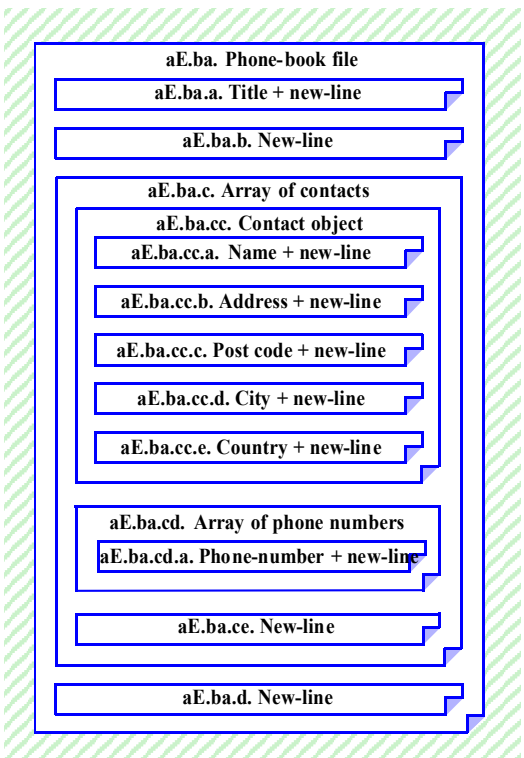


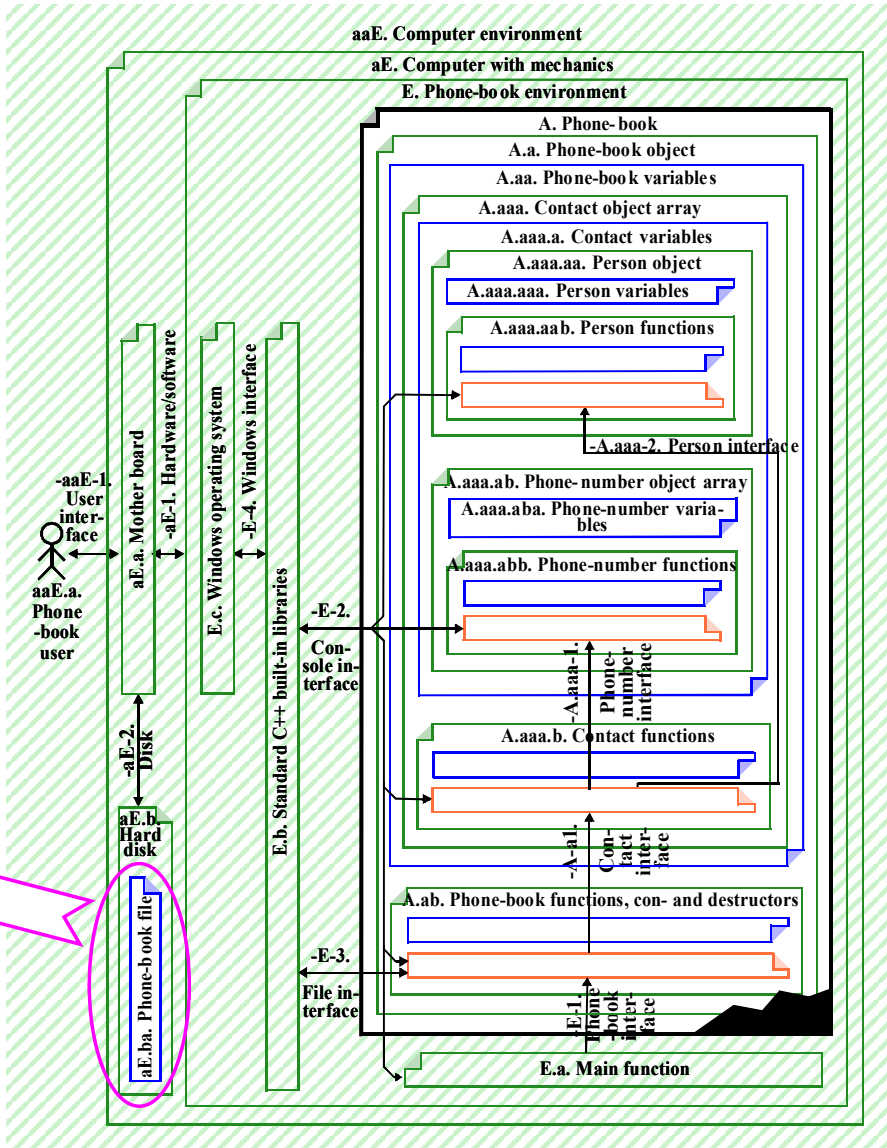
- Program file chart (kind of detailed physical architecture)



EXAMPLE Phonebook: Logical architectures

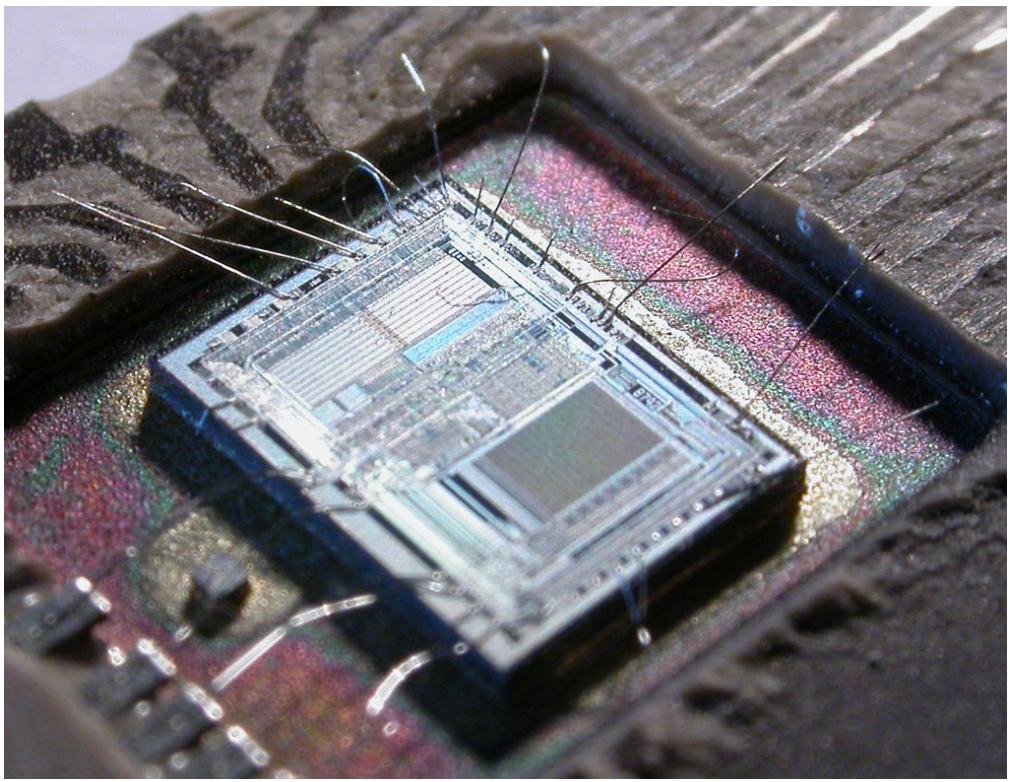
- Programs has hierarchical structures like mechanics and electronics
- Shown architecture assume object orientation
- Architecture shown of file storage





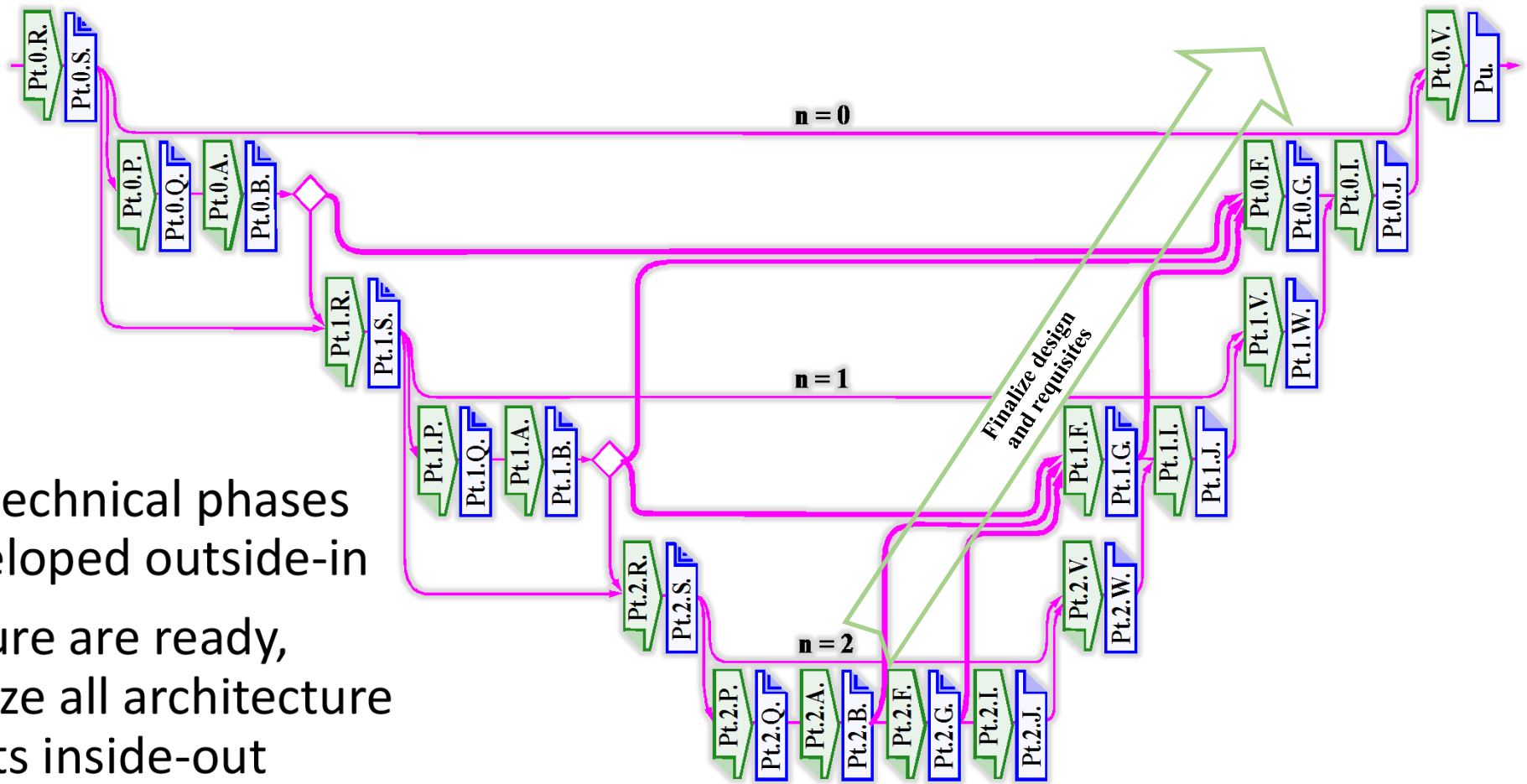
Finalize design

Programs lurking in silicon



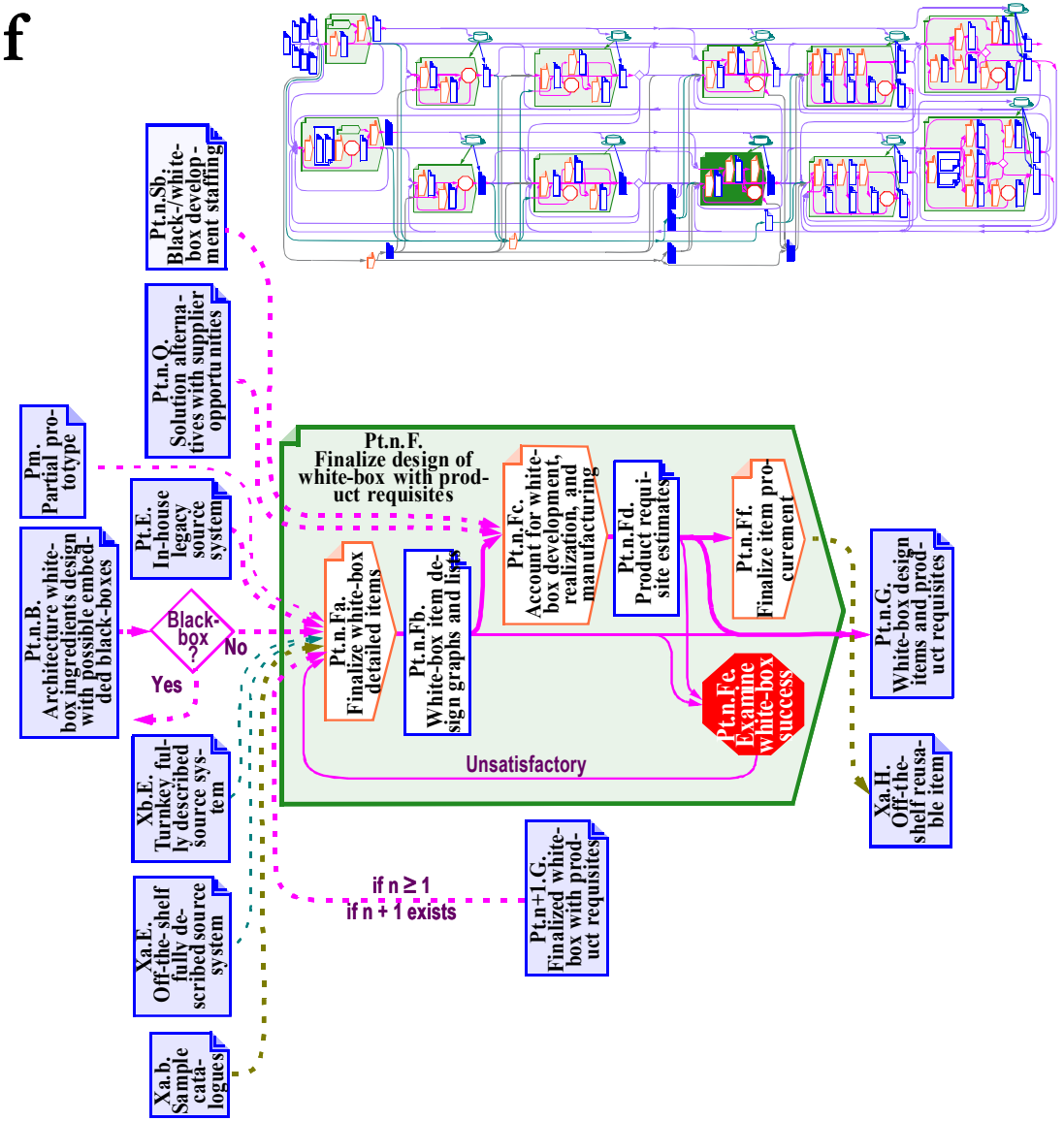
Finalize design inside-out

- Previous technical phases have developed outside-in
- Architecture are ready, now finalize all architecture ingredients inside-out



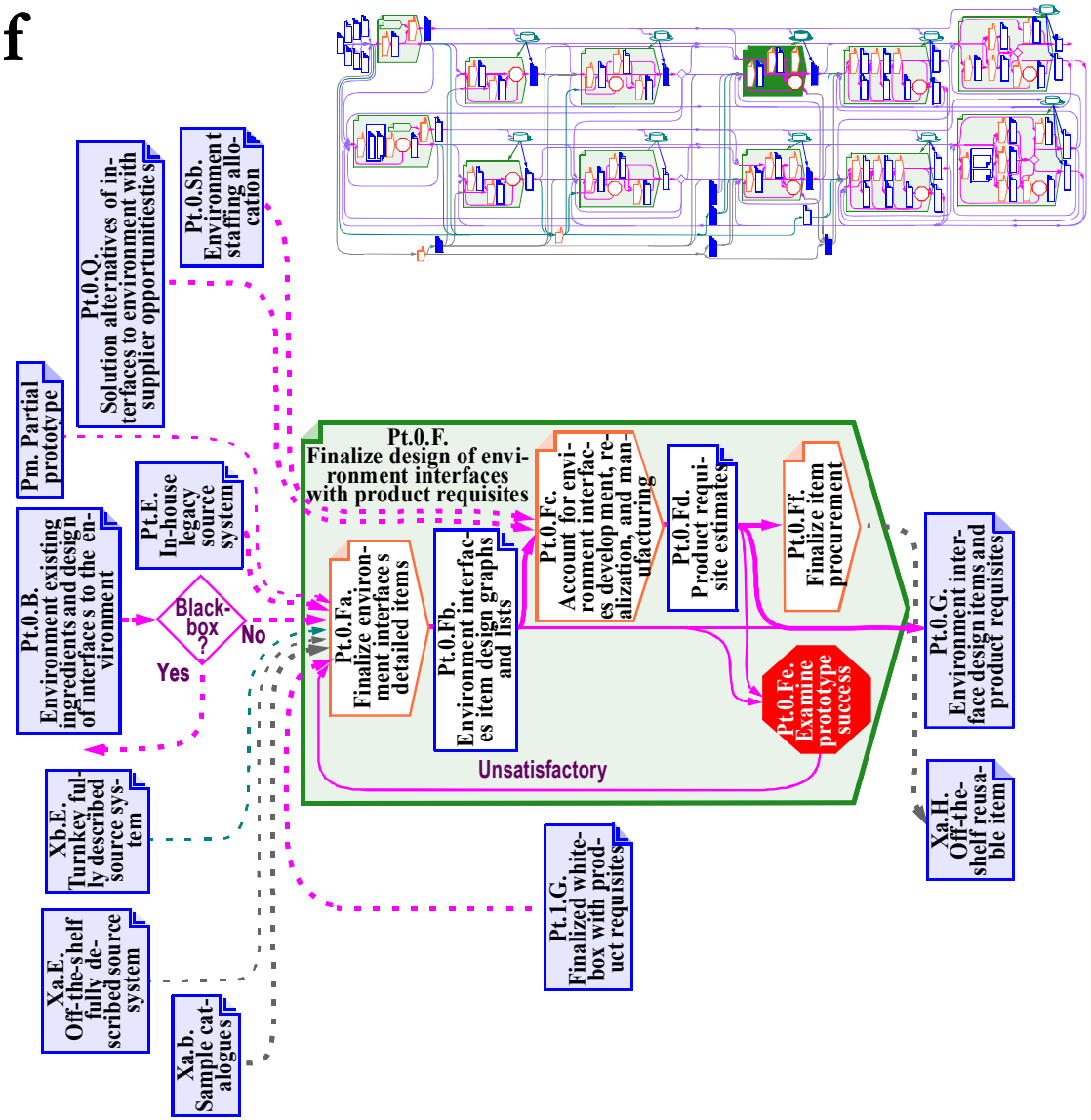
Pt.n.F. Finalize design of white-box with product requisites

- Finalize design ingredients for white-boxes on this level
- Anchor design on inwards finalized white-boxes
- Account for white-box requisites
- Finalize procurement of suppliers items
- Finalize design are quite similar for level 0 environment and level > 0 white-boxes



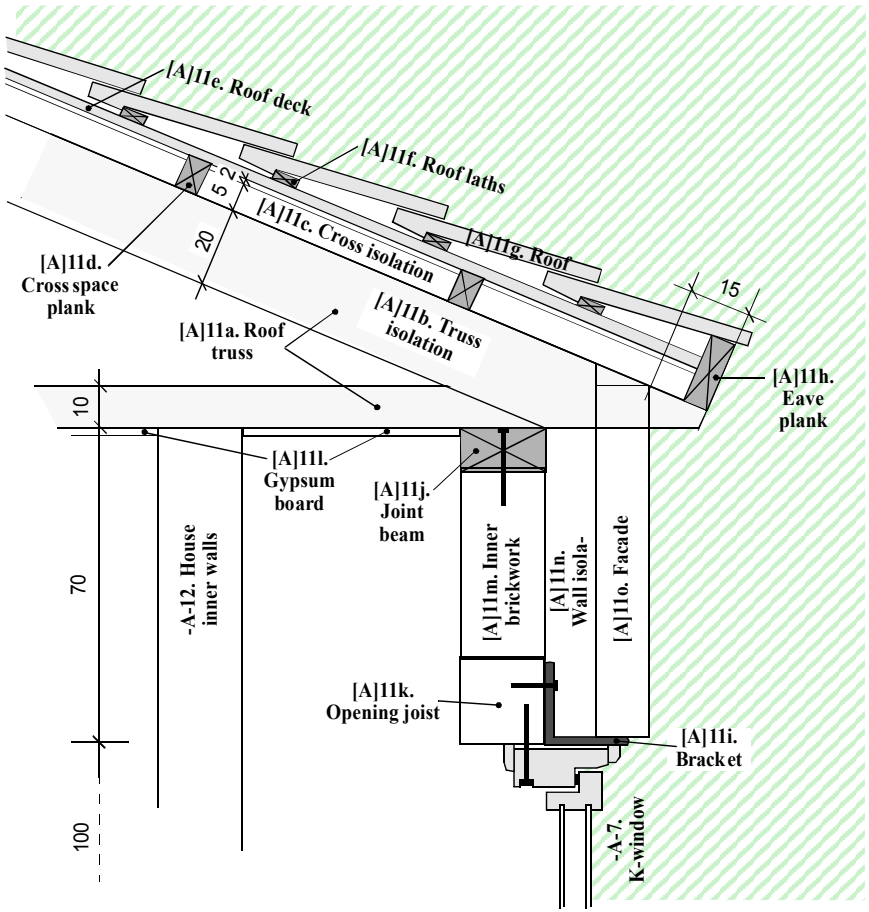
Pt.0.F. Finalize design of environment interfaces with product requisites

- Finalize design of interfaces to environment
- Finalized design performed inside-out
  - Interface anchor to finalized outmost white-box
- Account for development, realization and production of interface to environment
- Finalize item procurement



EXAMPLE House: Finalize white-box detailed items

- Finalize physical archi-tecture ingre-dients to become detailed design items



- Identify design items based on architecture ingredients

Item	Remark	Material and dimensions	Element or interface
i194. Roof truss	Prefabri-cated	Framework of spruce 5,10 wide x 2,0 m high	[A]11a. Roof truss
i195. Rock-wool	3GS	Board 60 x 120 x 20 cm	[A]11b. Truss isolation
i196. Rock-wool	3GS	Roll 20 m x 60 x 5 cm	[A]11c. Cross isolation
i197. Plank		Unplaned spruce 2 x 3 inch	[A]11d. Cross space plank
i198. Roof deck		0,5 x 5 inch,unplaned tongued spruce sur-plus plank	[A]11e. Roof deck
i199. Lath		1 x 1,5 inch unplaned spruce plank	[A]11f. Roof laths
i200. Plank		1,5 x 5 inch unplaned spruce plank	[A]11h. Eave plank
i201. Bracket		L-shape galvanized steel 15 x 15 cm	[A]11i. Bracket
i202. Plank		4 x 7 inch unplaned spruce plank	[A]11j. Joint beam
i203. Joist	Prefabri-cated	Concrete joist 18 x 18 cm	[A]11k. Opening joist
i204. Gypsum		Gypsum board 15 mm, 60 x 120 cm	[A]11l. Gypsum board
i205. Brick		20/m² hollow bricks 18 x 27 cm	[A]11m. Inner brickwork
i206. Mortar	Prefabri-cated	2600 kg fine concrete	
i207. Styro-foam		Styrofoam sheet 60 x 120 x 10 cm	[A]11n. Wall isolation
i208. Gravel		Sieved nature gravel 4 - 32 mm	[A]11p. Water barrier
i209. Footer	Concrete prefabri-cated	Moulded concreted	[A]11q. Foundation
i210. Rein-forcement bar		Iron bar Ø 15 mm	
i211. Macad-am		Macadam 16 - 32 mm	[A]11r. Macadam
i212. Pipe		Plastic perforated pipe	[A]11s. Drainage pipe
i213. Sole	Prefabri-cated	Fine concrete 10 cm	[A]11t. Sole
i214. Rein-forcement net		Bar Ø 8 mm, 120 x 240 cm, mesh 15 x 15 cm	

EXAMPEL Multiplication toy: Finalize white-box detailed program items

- Unstructured assembler program
  - Possible to jump anywhere
  - All variables available to change anytime
- Written in pseudo-code
  - Must be adapted to target microcontroller
- Efficient, but time consuming to write, read and verify
- Can be illustrated with flow chart

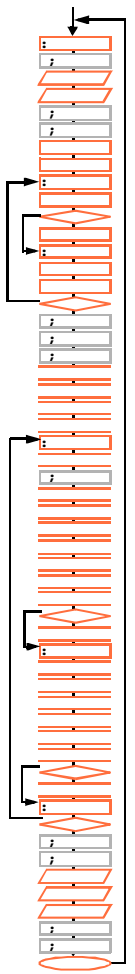
```
                ; Aa:2aa. Software program

                ; After start, the program counter is zero
                ; and execution begins at start: just below
                ; This program is written in generic assembler,
                ; which must be slightly adopted to the assembler
                ; set used by the target micro controller
                ; /* Write in 8051 assembler */
start:
; Get operands from settings
    MOV R1, P1 ; Move multiplicand on port 1 to register R1
    MOV R2, P2 ; Move multiplyer on port 2 to register R2

; perform the 4-bit * 4-bit multiplication
    MOV R3, 4  ; Use R3 as bit count, initiate with 4
    MOV R4, 0  ; Product will emerge in R4, initiate with 0
chkbit:
    SHR R2,    ; Right shift multiplyer, rightmost bit into carry
    JNC skip   ; Skip addition if carry is zero
    ADD R4, R1 ; Accumulate product in R4
skip:
    SHL R1,    ; Left shift multiplicand
    DEC R3     ; Decrement bit count
    JNZ chkbit ; Repeate if bits left

; perform the 8-bit binary to 3 digit BCD conversion
                ; Product resides in R4 as 8-bit binary value
    MOV R5, 0  ; BDC units will emerge in R5, initiate with 0
    MOV R6, 0  ; BDC tenth will emerge in R6, initiate with 0
    MOV R7, 0  ; BDC hunds will emerge in R7, initiate with 0
    MOV R8, 8  ; Use R8 as shift counter
nxtbit:
    SHL R4,    ; Left shift product, leftmost bit into carry
; rotate 1-bit left through unit, tenth and hunds, add 3 if >= 5
    RCL R5,    ; Left shift unit, carry into unit rightmost bit
    MOV R9, R5 ; Save unit in temporary R9
    ADD R9, 11110000 ; If bit 5 is 1, then carry->1 else carry->0
    RCL R6     ; Left shift thenth, carry into tenth rightmost bit
    AND R5, 0F ; Clear upper nibble of units
    MOV R9, R5 ; Save unit in temporary R9
    ADD R9, FB ; Add 251 to test if unit >= 5, if so carry -> 1
    JC skipunit  ; Skip adding if R9 < 5
    ADD R5, 3  ; Add 3 to unit
skipunit:
    MOV R9, R6 ; Save tenth in temporary R9
    ADD R9, 11110000 ; If bit 5 is 1, then carry->1 else carry->0
    RCL R7     ; Left shift hunds, carry into hunds rightmost bit
    AND R6, 0F ; Clear upper nibble of tenth
    MOV R9, R5 ; Save tenth in temporary R9
    ADD R9, FB ; Add 251 to test if tenth >= 5, if so carry -> 1
    JC skiptenth  ; Skip adding if R9 < 5
    ADD R5, 3  ; Add 3 to tenth
skiptenth:
    JNZ nxtbit : Repeat if bits left
;
; Send product to display
    MOV P3, R5 ; Move units to unit display
    MOV P4, R6 ; Move tenth to unit display
    MOV P5, R7 ; Move tenth to unit display
;
; Repeate everything
    JMP start  ; A jump is done by the microcontroller by loading
               ; the program counter with the loacation of the
```

Unstructured flowchart



Identify all kind of items

- For accounting, everything that has a development cost must become items
- See next slide

Aspect	Item					
Definition	All resources needed for product development, such as material, workforce, and equipment specified with desired granularity and formal enough to be traceable					
Aspect	**Design item** (see Table 10-2 below)			**Role item**		**Any kind of items**
Definition	Material (including linkable programs) specified with desired granularity formal enough to be orderable			Accomplished and estimated work specified with desired granularity and formal enough to be traceable		Anything specified with desired granularity
Synonym	Bill of material (BOM).					
Aspect	**In-house design item**	**Supplier design item**		**In-house role item**	**Provider role item**	**For appropriate accounting**
Definition		Design developed by suppliers				
Aspect		**Turn-key item**	**Off-the-shelf item**			
Definition	Design developed by in-house workforce	Custom-made design developed to fit very well into own design	Predeveloped reusable components	Development, realization, and manufacturing in-house workforce	Development, realization, and manufacturing hired workforce	Partitioning of anything too big for desired accounting granularity (see ch. 10.23, p. 560)

Account all production requisites

- Items to be accounted:
  - Estimated manufacturing material requisites cost
  - Estimated manufacturing work requisites hourly/fixed/rental cost
  - Estimated manufacturing equipment requisites purchase/rental cost
  - Estimated realization material requisites cost
  - Estimated realization work requisites hourly/fixed/rental cost
  - Estimated realization equipment requisites purchase/rental cost
  - Estimated development material requisites cost
  - Estimated development work requisites hourly/fixed cost
  - Estimated development equipment requisites purchase/rental cost
  - Summed/accumulated requisites estimates

EXAMPLE House: Account for material (contimue)

- All items are accounted
- Material is accounted and summarized

Nº	Unit	€/unit	€	Item	Product	Supplier	Interface
22	Piece	410	9 020	i194.Roof truss		SU15. AllBau	[A]11a. Roof truss
132	Sheet	3.8	502	i195.Rockwool		SU19. IsoBau's suppliers	[A]11b. Truss isolation
8	Roll	38	304	i196.Rockwool		SU19. IsoBau's suppliers	[A]11c. Cross isolation
160	meter	1.6	256	i197.Plank		SU20. Local suppliers	[A]11d. Cross space plank
950	meter	0.28	266	i198.Roof deck		SU20. Local suppliers	[A]11e. Roof deck
260	meter	0.13	34	i199.Lath		SU20. Local suppliers	[A]11f. Roof laths
27	meter	1.9	51	i200.Plank		SU20. Local suppliers	[A]11h. Eave plank
7	meter	8.5	60	i201.Bracket	IronWare	SU15. AllBau	[A]11i. Bracket
27	meter	11	297	i202.Plank		SU20. Local suppliers	[A]11j. Joint beam
5	meter	17.2	86	i203.Joist		SU15. AllBau	[A]11k. Opening joist
2	meter	17.2	34	i203.Joist		SU15. AllBau	-A-12. House inner walls
1600	Piece	1.2	1 920	i205.Brick	Bricket	SU15. AllBau	[A]11m. Inner brickwork
5	m³	67	335	i206.Mortar		SU20. Local suppliers	[A]11m. Inner brickwork
500	Piece	1.2	600	i205.Brick	Bricket	SU15. AllBau	-A-12. House inner walls
2	m³	67	134	i206.Mortar		SU20. Local suppliers	-A-12. House inner walls
50	Sheet	5.8	290	i204.Gypsum		SU20. Local suppliers	[A]11l. Gypsum board
112	Sheet	0.7	78	i207.Styrofoam		SU19. IsoBau's suppliers	[A]11n. Wall isolation
7	m³	25	175	i208.Gravel		SU20. Local suppliers	[A]11p. Water barrier
15	m³	70	1 050	i209.Footer		SU20. Local suppliers	[A]11q. Foundation
420	meter	0.3	126	i210.Reinforcement bar	IronWare	SU15. AllBau	[A]11q. Foundation
10	m³	57	570	i211.Macadam		SU20. Local suppliers	[A]11r. Macadam
36	meter	5	180	i212.Pipe		SU20. Local suppliers	[A]11s. Drainage pipe
7	m³	56	392	i213.Sole		SU15. AllBau	[A]11t. Sole
36	Piece	14	504	i214.Reinforcement net	IronWare	SU15. AllBau	[A]11t. Sole
4700	Piece	0.043	202	i232.Brick	Bricket	SU15. AllBau	[A]11o. Facade
5	m³	76	380	i233.Mortar		SU20. Local suppliers	[A]11o. Facade
600	meter	0.2	120	i234.Reinforcing	IronWare	SU15. AllBau	[A]11o. Facade
160	Piece	0.14	22	i235.Hooks		SU15. AllBau	[A]11o. Facade
1260	Piece	4.7	5 922	i236.Tiles	Bricket	SU15. AllBau	[A]11g. Roof tile
1	Piece	560	560	i237.Window		SU17. Pa&Pp	-A-7. K-window
1	Piece	320	320	i238.Window		SU17. Pa&Pp	-A-8. T-window
1	Piece	872	872	i239.Door		SU17. Pa&Pp	-A-9. Front door
1	Piece	541	541	i240.Door		SU17. Pa&Pp	-A-6. Door section
2	Piece	330	660	i241.Window		SU17. Pa&Pp	-A-6. Door section
1	Piece	518	518	i215.Door		SU17. Pa&Pp	-A-13. E-door
1	Piece	530	530	i216.Door		SU17. Pa&Pp	-A-14. Lock door
1	Piece	1 680	1 680	i218.Stair		SU17. Pa&Pp	-A-16. Stairway
1	Piece	320	320	i219.Wicket		SU17. Pa&Pp	-A-17. S-wicket
1	Piece	360	360	i220.Wicket		SU17. Pa&Pp	-A-18. L-wicket
8	meter	7	56	i221.Pipe		SU14. Builder's suppliers	-A-1. Warm water
8	meter	7	56	i222.Pipe		SU14. Builder's suppliers	-A-2. Cold water
80	meter	7.2	576	i223.Hose		SU14. Builder's suppliers	-A-3a. K-heating
90	meter	7.2	648	i224.Hose		SU14. Builder's suppliers	-A-3b. D-heating
30	meter	7.2	216	i225.Hose		SU14. Builder's suppliers	-A-3c. E-heating
30	meter	7.2	216	i226.Hose		SU14. Builder's suppliers	-A-3d. T-heating
40	meter	9.7	388	i228.5xCable		SU18. Elert's suppliers	-A-4a. K-Wiring
48	meter	7.1	341	i229.3xCable		SU18. Elert's suppliers	-A-4b. O-Wiring
8	meter	6.7	54	i230.Drain pipe		SU14. Builder's suppliers	-A-5. Sewer
3	meter	14	42	i231.Coil pipe		SU14. Builder's suppliers	-E-1. Air outlet
			32 864				

(Continued) EXAMPLE House: Account for development, realization and accumulate

- Development >
- Realization >
- Accumulate >

h	€/h	€	Item	Provider	Process schedule
		0	i25. House user	PR.10. Proprietor	Pt:2:2.1.R. Refine house requirements from environment restrictions and ensure house development staffing
30	50	1 500	i26. Architect	PR.11. Robber HB	Pt:2:2.1.P Predetermine house solutions and supplier opportunities
100	40	4 000	i27. Architect	PR.11. Robber HB	Pt:2:2.1.A. Satisfy house requirements by decomposing house black-box into white-box design containing room black-boxes
20	40	800	i28. Artistic architect	PR.15. Homey Inc	Pt.n.Ae. Layout physical architecture
30	40	1 200	i29. House layout specialist	PR.15. Homey Inc	
8	50	400	i30. Electrician	PR.12. Elert AB	Pt:2:2.1.A. Satisfy house requirements by decomposing house black-box into white-box design containing room black-boxes
150	30	4 500	i31. Calculation engineer	PR.11. Robber HB	Pt:2:2.1.F. Finalize house elements and interfaces with product requisites
24	40	960	i32. Electrician	PR.12. Elert AB	Pt:2:2.1.F. Finalize house elements and interfaces with product requisites
40	50	2 000	i48. Quality inspector	PR.10. Proprietor	Pt:2:2.1.V. Verify house white/black-box
		15 360			

Item	Cost €
Rooms and machinery	63 830
Stairway nook realization material	5 934
Stairway nook realization work	3 680
Loft realization material	1 940
Loft realization work	2 340
House realization material	32 864
House realization work	74 680
House development work	15 360
House, room and machinery sum	**200 628**

h	€/h	€	Item	Provider	Interfaces
40	80	3 200	i33. Labourer	SU14. Builder's suppliers	-E-3. Sewage -E-2. Water
96	40	3 840	i47. Electrician	SU18. Elert's suppliers	-E-4. Electricity
48	150	7 200	i35. Pipe-fitter	SU14. Builder's suppliers	-A-5. Sewer -A-2. Cold water -A-1. Warm water [A]11s. Drainage pipe
24	50	1 200	i36. Authorized gas plumber	SU14. Builder's suppliers	-E-5. Gas
140	40	5 600	i37. Founder	SU14. Builder's suppliers	[A]11q. Foundation [A]11t. Sole
40	140	5 600	i34. Excavator with driver and 1 labourer	SU14. Builder's suppliers	[A]11r. Macadam [A]11p. Water barrier
56	30	1 680	i39. Plumber	SU14. Builder's suppliers	-A-3a. K-heating -A-3b. D-heating -A-3c. E-heating -A-3d. T-heating -A-3e. B-heating -A-3d. T-heating
448	30	13 440	i40. Bricklayer	SU14. Builder's suppliers	[A]11m. Inner brickwork, [A]11k. Opening joist, [A]11j. Joint beam
344	30	10 320	i41. Facade bricklayer	SU14. Builder's suppliers	[A]11i. Bracket [A]11n. Wall isolation [A]11o. Facade
16	40	640	i38. Tin-smith	SU14. Builder's suppliers	-E-1. Air outlet
48	30	1 440	i42. Chimney mason	SU14. Builder's suppliers	-E-2. Chimney
48	35	1 680	i43. Carpenter	SU14. Builder's suppliers	-A-9. Front door -A-10. Back door -A-8. T-window -A-7. K-window -A-6. Door section
124	30	3 720	i44. Cabinet maker	SU14. Builder's suppliers	-A-13. E-door -A-14. Lock door -A-15. Open arch -A-16. Stairway -A-17. S-wicket -A-18. L-wicket [A]11l. Gypsum board
36	140	5 040	i46. Woodworker	SU14. Builder's suppliers	[A]11a. Roof truss
252	40	10 080	i45. Roofer	SU14. Builder's suppliers	[A]11b. Truss isolation, [A]11d. Cross space plank, [A]11c. Cross isolation[A]11e. Roof deck, [A]11f. Roof laths, [A]11h. Eave plank
		74 680			

EXAMPLE Phonebook: Account for development

- Programs can be time- and man-consuming to develop
- Accounting granularity need to be more detailed
  - Modularize programs into items
  - Categorize developers into items
- Account in many dimensions

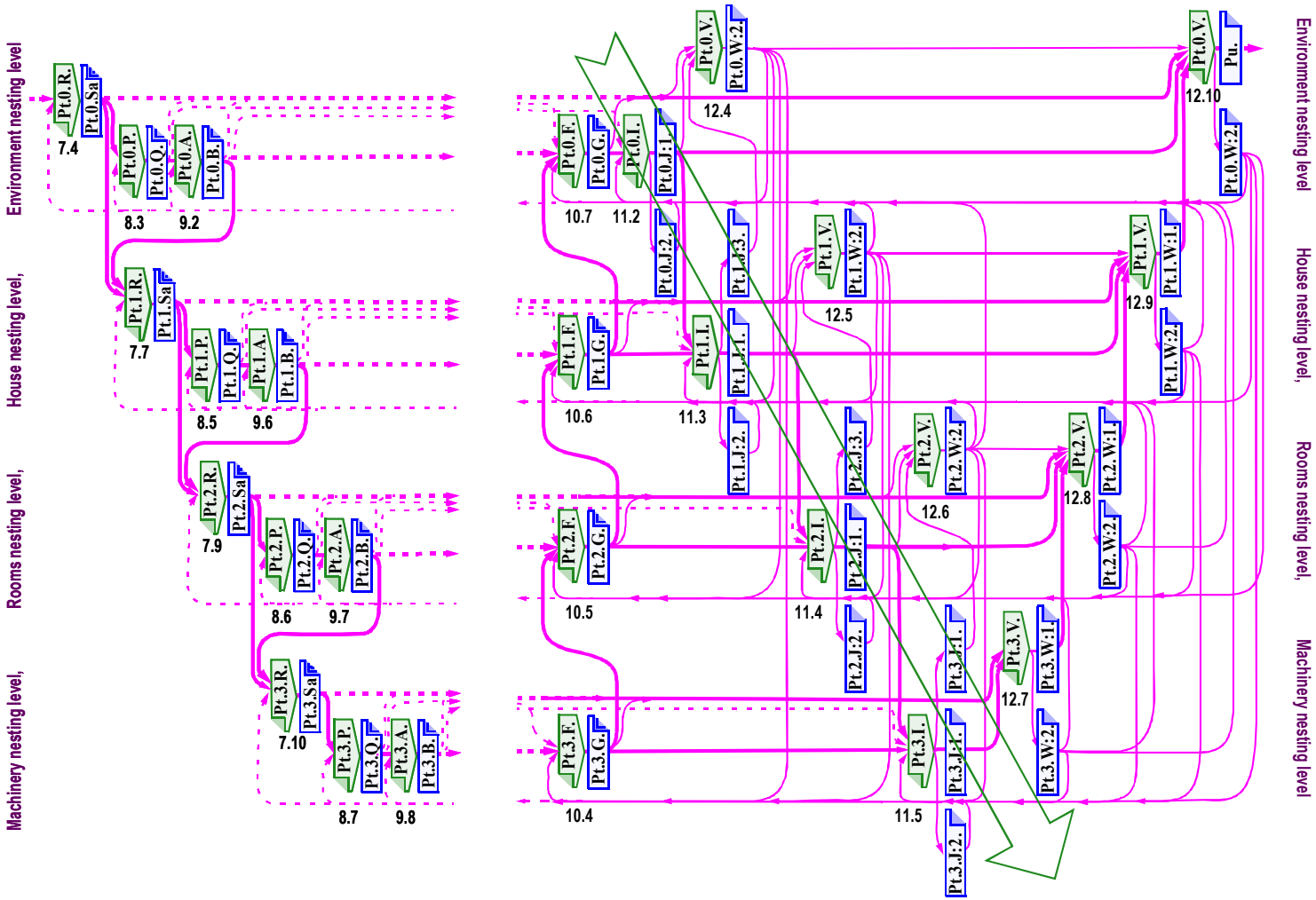
Pt.n.Fa. Finalize white-box detailed items									
i25. Architect programmer			i26. Object-oriented programmer			i27. General C++ programmer			
#	€/h	€	#	€/h	€	#	€/h	€	File item
48	45	2 160				8	60	480	i106. Header file ph_book_com_shapes.h
						15	60	900	i104. Header file ph_book_restrict.h
						5	60	300	i105. Header file ph_book_alias_etc.h
12	45	540							i108. Header file support_f_shape.h
						30	60	1 800	i107. Source file ph_book_dyn_inst.ccp
						15	60	900	i109. Source file support_f.cpp
20	45	900	10	50	500				i110. Header file ph_book_c_shape.h
						32	60	1 920	i111. Source file ph_book_c.ph_-book_c.ccp
						25	60	1 500	i112. Source file ph_book_c.key_add_-cont_f.cpp
						10	60	600	i113. Source file ph_book_c.disp_all_-cont_f.cpp
						32	60	1 920	i114. Source file ph_book_c.delete_-cont_f.cpp
						30	60	1 800	i115. Source file ph_book_c.~ph_-book_c.cpp
8	45	360	20	50	1 000				i116. Header file cont_c_shape.h
						36	60	2 160	i117. Source file cont_c.key_cont_f.cpp
						12	60	720	i118. Source file cont_c.disp_cont_f.cpp
						22	60	1 320	i119. Source file cont_c.read_cont_f.cpp
						20	60	1 200	i120. Source file cont_c.write_cont_f.cpp
8	45	360	16	50	800				i121. Header file pers_c_shape.h
						8	60	480	i122. Source file pers_c.key_pers_f.cpp
						8	60	480	i123. Source file pers_c.disp_pers_f.cpp
						15	60	900	i124. Source file pers_c.read_pers_f.cpp
						25	60	1 500	i125. Source file pers_c.write_pers_f.cpp
12	45	540	16	50	800				i126. Header file ph_numb_c_shape.h
						6	60	360	i127. Source file ph_numb_c.key_ph_-numb_f.cpp
						4	60	240	i128. Source file ph_numb_c.disp_ph_-numb_f.cpp
						14	60	840	i129. Source file ph_numb_c.read_ph_-numb_f.cpp
						10	60	600	i130. Source file ph_numb_c.write_-numb_f.cpp
Sum		4 860			3 100			22 920	

Integrate white-box

Will the items fit a prototype ?

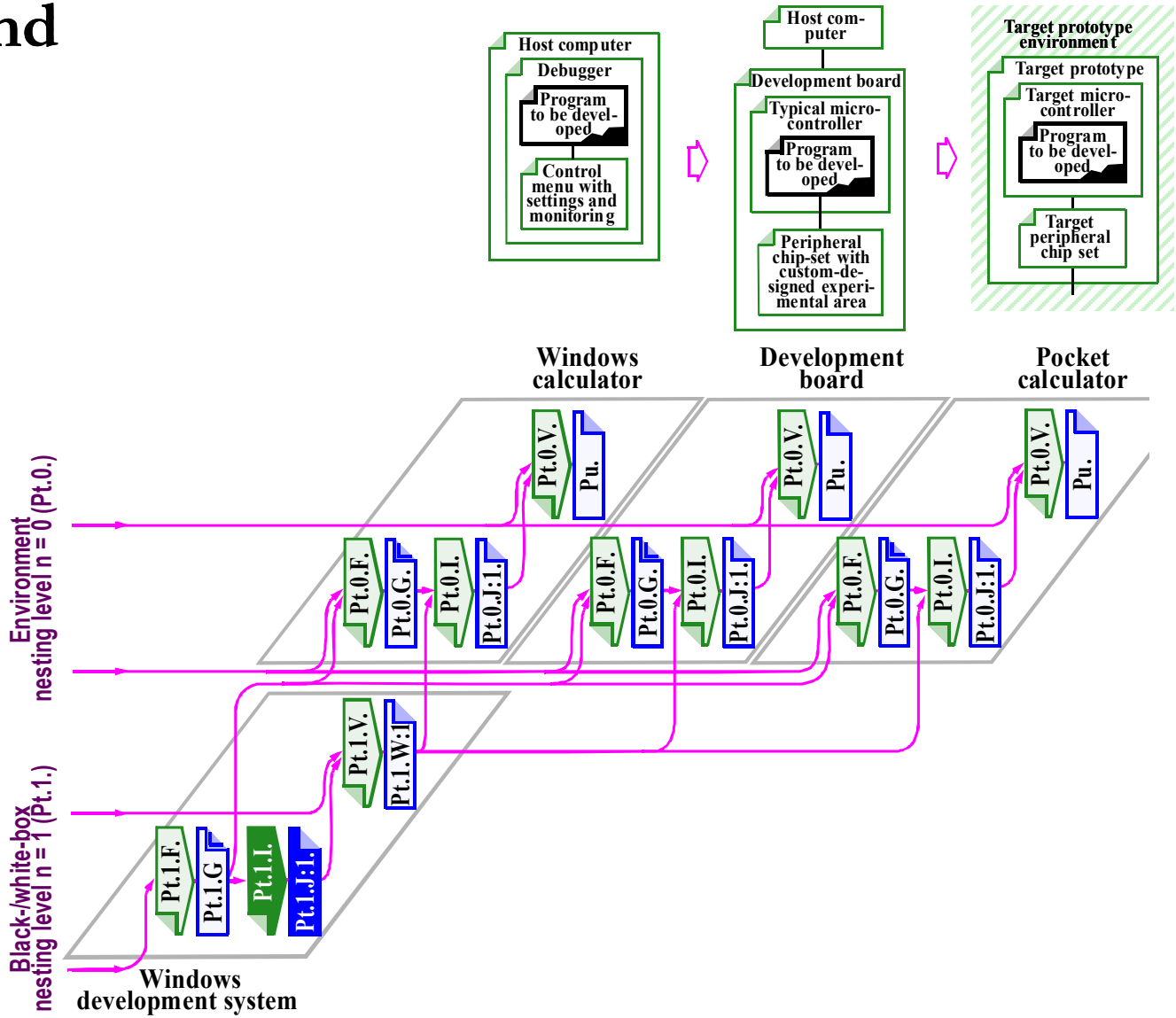
Invasive integration with partial and full verification

- Integrate outside-in
  - Make embedded black-boxes to fakes
  - Verify partially with fake
  - Remove fake and invade
- Verify fully inside-out
  - When all has been integrated

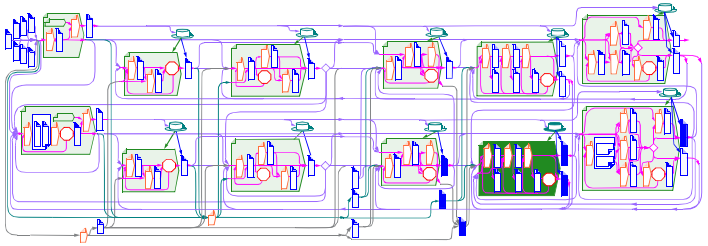


Develop in host and port to target

- Porting a product from development system to target in field can be a too challenging lap
- Stepwise porting
  1. Develop and verify outmost white-box in host debugger
  2. Port to host environment
  3. Port to development board having target processor
  4. Port to target environment

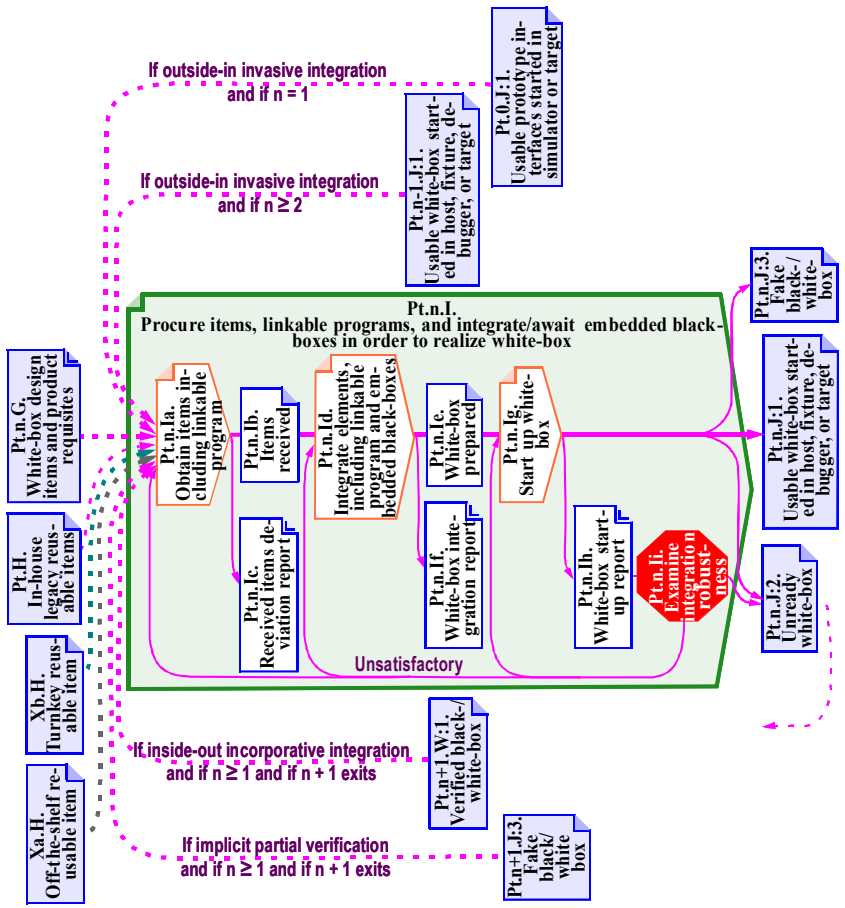


Pt.n.I. Procure items, linkable programs, and incorporate/invade embedded black-boxes in order to realize white-box

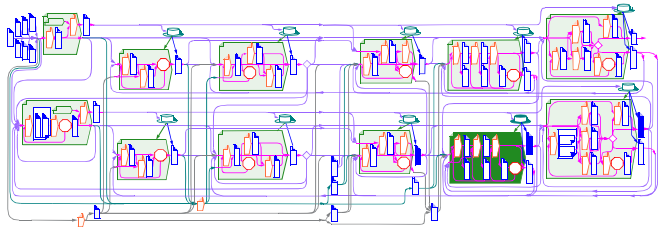


- Receive items for white-box from suppliers
- Realize white-box and integrate
  - By incorporate embedded black-boxes

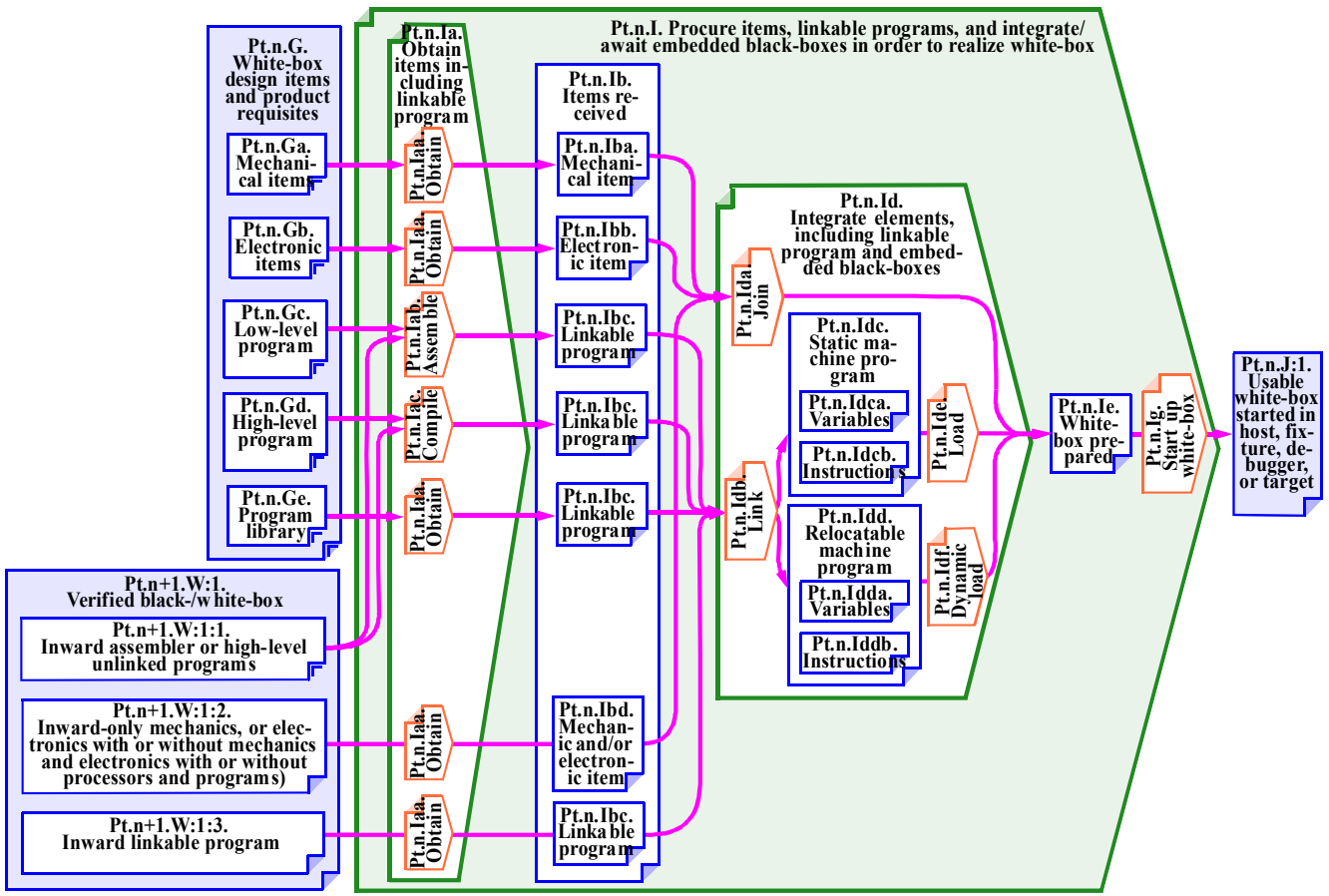
  or
  - By invading embedded black-boxes
- Start-up white-box in host, debugger, fixture or target



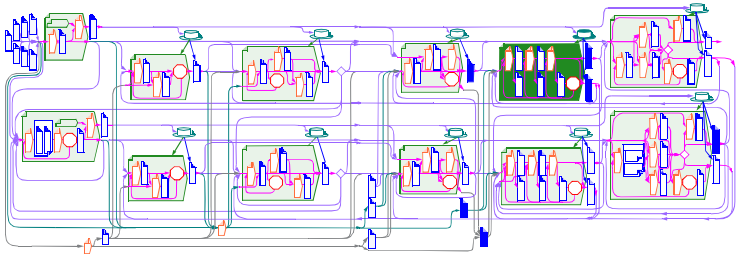
Expanding the Pt.n.I. schedule



- Expanded Pt.n.I. schedule showing integration of different technologies
- Integration of embedded
  - Assembler and high level programs
  - Mechanics and electronics possibly containing processors with programs
  - Linkable programs

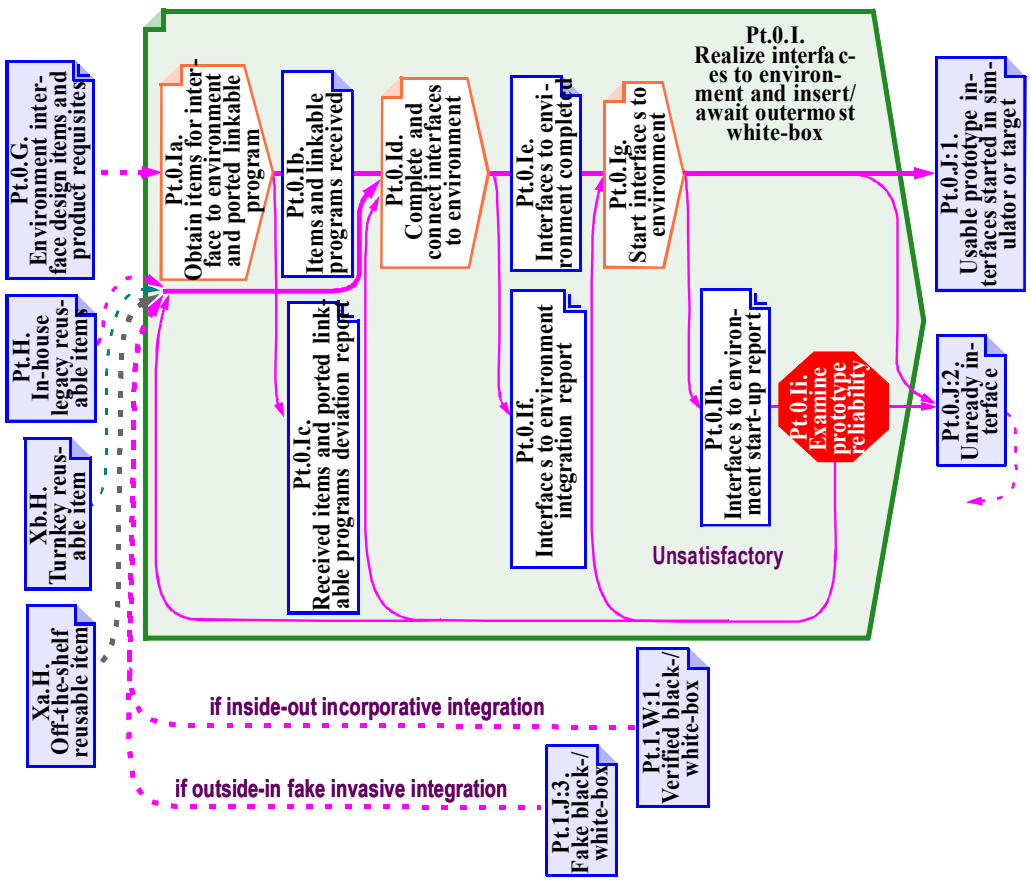


Pt.0.I. Realize interfaces to environment and incorporate/ invade outermost white-box



- Receive items for interface to environment
- Realize interface to environment and integrate
  - by incorporate outermost black-box

  or
  - by invade outermost black-box
- Start up prototype in environment



EXAMPLE Multiplication toy: Reports

- Deviation report

Volume					1 000		100 000		
Deviation	Item	Sourcing	Producer	Pieces	€/piece	Cost € / toy	€/piece	Cost € / toy	Id Arch
DR.1. Difficult to solder on printed board, must be improved	Rotor switch	Gross Electro	Pointer	2	0.4	0.8	0.2	0.4	Ab.
DR.2. To be phased out by supplier	LED-segment, no dp.	Gross Electro	Texx	3	0.3	0.9	0.2	0.6	Ad.
DR.3. All ordinary bins too high. Order specialized immediately.	Battery bin	Gross Electro	Any	1	0.2	0.2	0.12	0.12	Ae.
DR.4. Up to 4 month delivery time. In the meantime, use manually produced fronts, backs and flaps for prototypes.	Pressed front housing	Inhouse	Inhouse	1	0.2	0.2	0.1	0.1	A-2c.
	Front housing tool	In depth press	In depth press	1 per 50 000	5	5	0.1	0.1	A-2c.
	Back housing	Inhouse	Inhouse	1	0.1	0.1	0.06	0.06	A-2d.
	Back housing tool	In depth press	In depth press	1 per 50 000	5	5	0.1	0.1	A-2d.
DR.5. Produced by same tool as above housing	Cut battery flap	Inhouse	Inhouse	1	0.025	0.025	0.02	0.02	A-2f.
DR.6. Maximum capacity 4000 pieces / week	Display window	Gross Electro	Desiger	1	0.02	0.02	0.015	0.015	A-2e.
DR.7. There might be better controller in this huge family	PIC16F57	Pick-it	MiChip	1	0.7	0.7	0.4	0.4	Aa:2a.
DR.8. Up to 2 month delivery time. Use expensive fast produced PCB for prototypes	PCB	Inhouse	PCB inc.	1	0.4	0.4	0.2	0.2	-A-1:2
DR.9. Add probe area for automated tests	PCB preparation	Inhouse	Inhouse	1	0.4	0.4	0.01	0.01	-A-1:2

- Integration report
  - Based on interfaces

Fulfilments and shortcomings	Contained interfaces	
IR.1. The circuit board were equipped without problems.	-A-1:2Printed board	
IR.2. Riveted on selectors axis	[A]2.Housing	A-2a.Dial knob
IR.3. Soldered to the printed circuit board. Note, there are components on both sides.		A-2b.On-off Switch
IR.4. Manually produced front mounted		A-2c.Toy front housing
IR.5. Manually produced back mounted		A-2d.Toy back housing
IR.6. Manually produced Plexiglas mounted		A-2e.Display window
IR.7. Manually produced flap mounted		A-2f.Battery flap
IR.8. Ordinary screw temporarily used		A-2g.Flap screw
IR.9. Selector soldered with some problem. Machine soldering not possible.	-A-3.Operand bus	
	-A-4.Operand bus	
IR.10. Presentation components mounted without problems on the printed circuit board printed bus	-A-5.Product bus	
IR.11. The power is distributed on all circuit board printing.	-A-6.Power feed	
IR.12. Loading the machine code to micro controller. The protocol is rather tricky and a specialist from the supplier was called in to get it right first time.	-Aa:2-1.In circuit programmable port	

- Start-up report

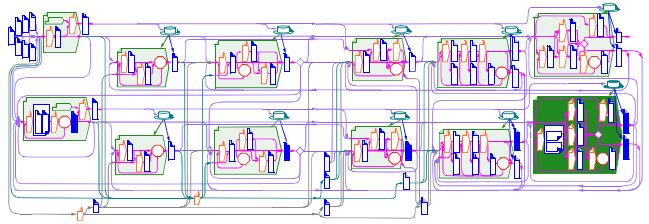
Accomplishments and obstacles	Troubling interface
SR.1. The power switch is set to on, and the multiplication toy display illuminates nicely, and response nicely when changing the rotating switches. Multiplication incorrect, but ready for verification.	A. Multiplication toy with micro controller multiplier

Verify white- and black-box

Can even be automated

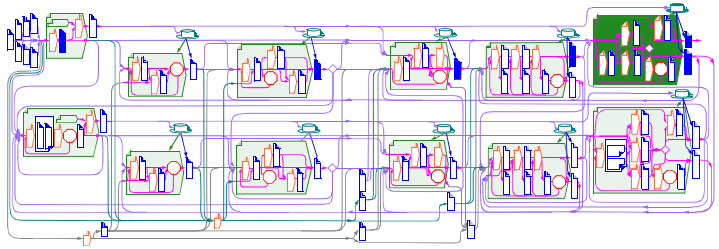
Pt.n.V. Verify black-/white-box

- Types of verifications
  - Explicit verification inside-out
  - Partially verification outside-in
  - Fully implicit verification inside-out
- Develop white- and black-box test-cases for fixtures and debugger
- Verify
  - White-box
  - White/black-box
  - Black-box
- Locate and correct failures
- Reverify until no more failures
- Prepare manual table of content



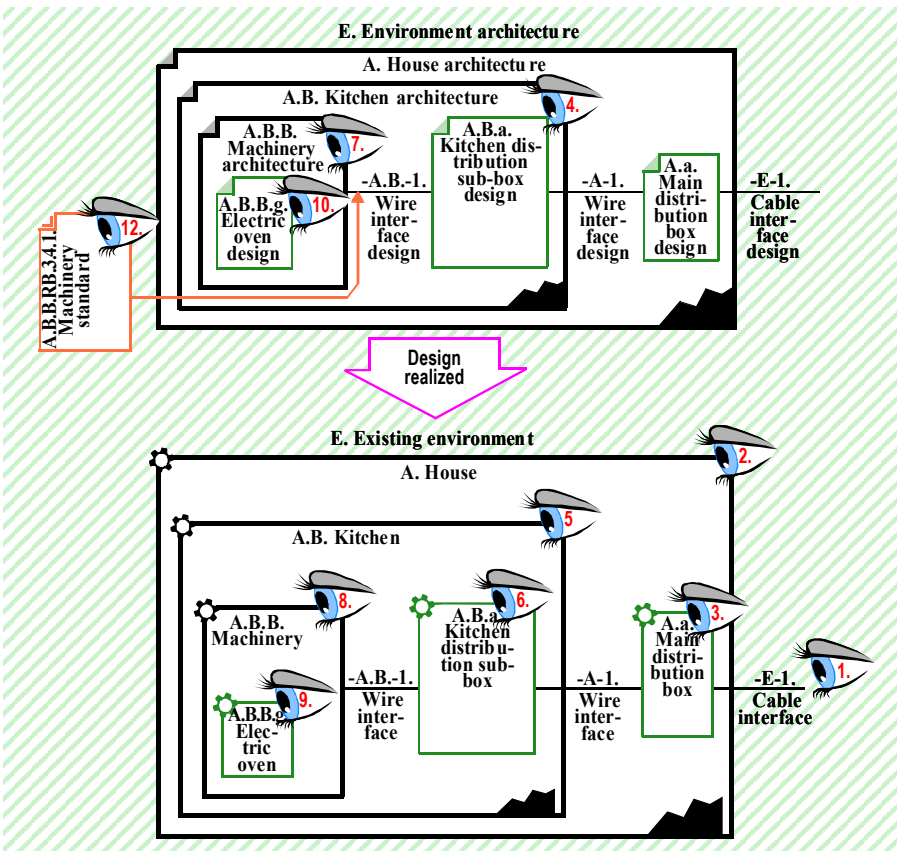
Pt.0.V. Verify prototype in environment

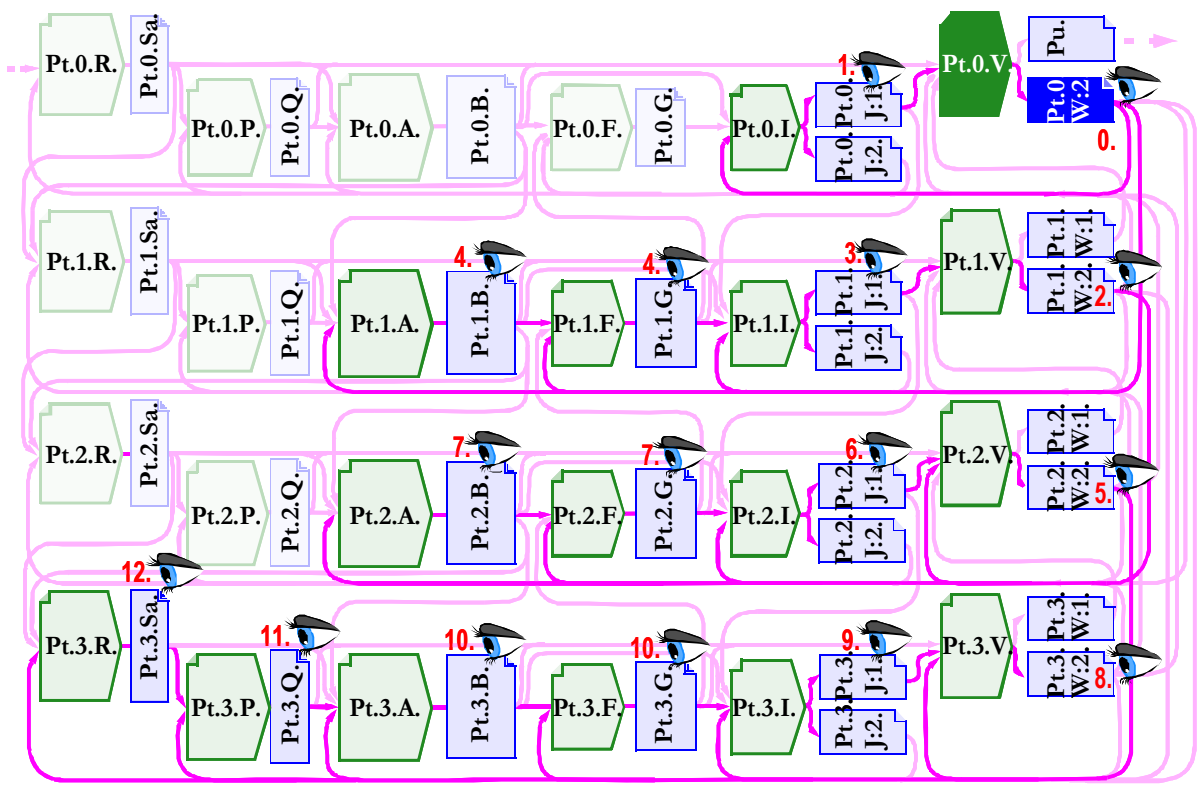
- Develop field test cases for target or simulator
- Verify prototype
  - In simulator
  - In target
- Locate and correct failures
- Reverify until no more failures
- Complete manual



Find a failure localization by tracing upstream

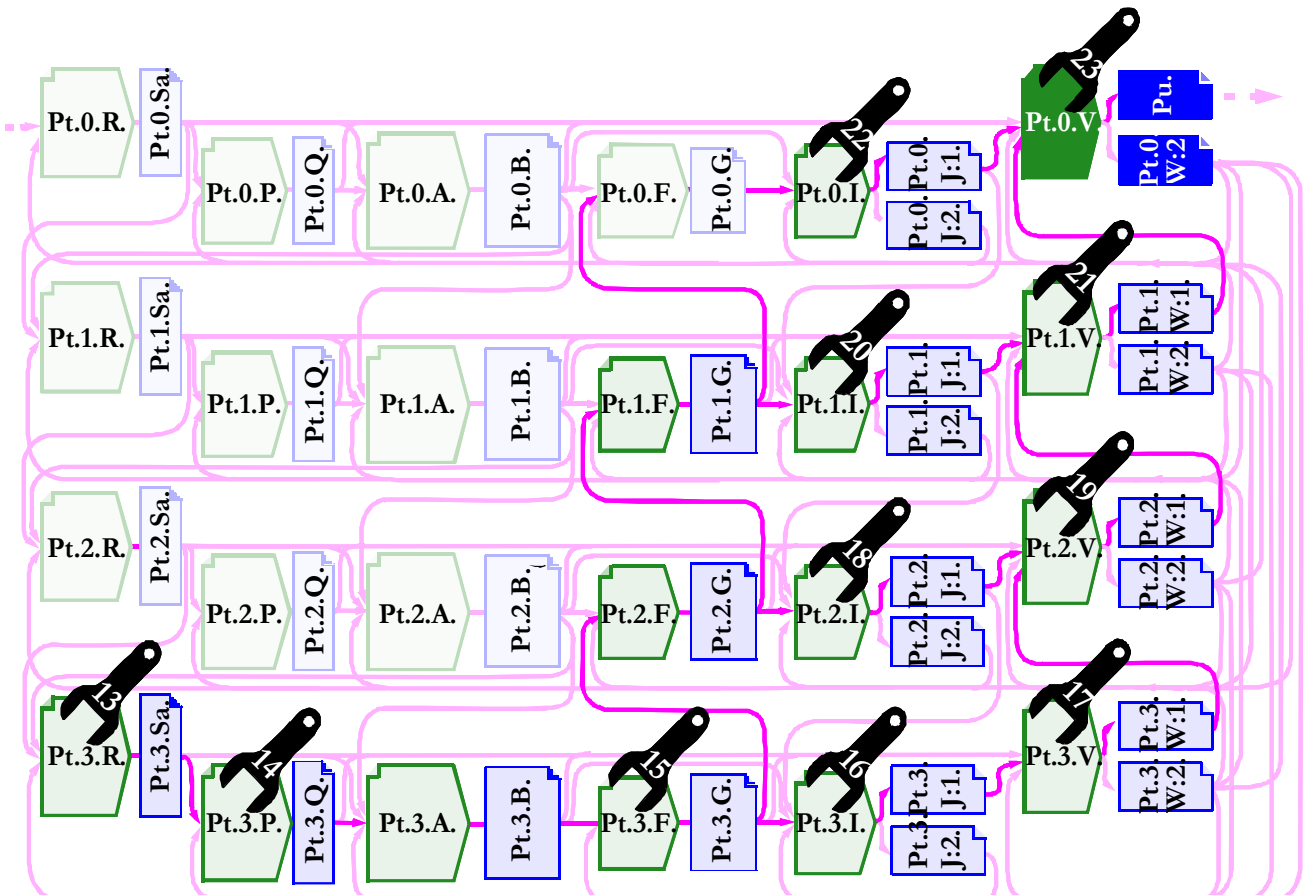
- Assume failure were found on interface -E-1 in Pt.0.V. activities
- Assume failure is located in requirement A.B.B.RB.3.4.1.





Eliminate found failure by tracing downstream

- When primary failure is found
  - Correct this failure
  - Trace downstream from this failure and adjust to bring everything in consistence
- During downstream tracing also localize eventual secondary failures caused by the primary failure
  - Correct and adjust like for the primary failure



EXAMPLE Phonebook: White- and black-box test cases

- White-box test cases

White-box design check	Expected appearance with pass-fail criteria	Restriction requirement
CH.1. Check which compiler is used.	XP.1. Pass, if used compiler is widely available, considered trustworthy and inexpensive.	A.RR.1.1. The phone-book program shall be written for a standard windows computer.
CH.1. Let a senior designer check the structure of the program.	XP.1. Pass, if the program is developed according to designed architecture and if it is highly modularized in a competent way.	A.RR.2.1. First phone-book program shall use simple text oriented console interaction.
CH.1. Start the phone-book, add a contact and terminate the program.	XP.1. Pass if the added contact is saved in a phone-book file.	A.RR.3.1. The program shall load phone-book data at startup and save it on program termination.
CH.1. Let a programmer check the program.	XP.1. Pass, if the selected language is object oriented.	A.RR.4.1. Chose an object oriented development system, and keep together user interaction with used objects.
CH.1. Check the program code	XP.1. Pass, if the program uses fixed arrays for variable storages.	A.RR.5.1. The first phone-book program version shall use fixed size data storage
CH.1. Check the program code	XP.1. Pass if specified restrictions are applied in the program code.	A.RR.5.2. MAX_CONTACTS = 128
• Compared to maximum length, key in person name with: CH.1. one character less than maximum CH.2. equal number of characters CH.3. one character more	XP.1. Pass, if oversized person data and phone numbers get truncated at their max length and if the program continues to execute normally and if these data keep after termination and restart.	A.RR.5.4. MAX_PERS_LEN = 128
• Compared to maximum length, key in phone numbers with: CH.4. one character less CH.5. equal number of characters CH.6. with one character more		A.RR.5.5. MAX_NUMB_LEN = 32

- Black-box test cases

Black-box stimulus inject	Expected response with pass-fail criteria	Behaviour requirement
• Use a phone-book file which has various number of existing contactsStart the phone-book program, and append one contact. CH.1. Phone-book file correct, but without contacts. CH.2. Phone-book file approximately half full. CH.3. Phone-book file still having place for one more contact • For each of the above ways of registering contacts, also register phone numbers in the five various ways CH.4.a. One phone number CH.4.b. Half full of phone numbers CH.4.c. Next to full of phone numbers CH.4.d. Full of phone numbers	XP.1. Pass, if the appended contact and phone numbers are present in the phone-book at a listing of contacts.	A.RB.6.S.2.U.1. Append contact to phone book normal case
CH.1. Use a phone-book file which has no more place for contacts. Start the phone-book program, and append one contact.	XP.1. Pass, if the program display an error message, and if a listing doesn't contain anything from the appended contact, and if the program still execute normally.	A.RB.6.S.2.X.2. Append contact to full phone book exception case
CH.1. Use a phone-book file which is half full of contacts. Start the phone-book program. CH.2. Append one contact and key in the maximum number of phone numbers. CH.3. Try to key in yet one more phone number.	XP.1. Pass, if the program display an error message, and if a listing doesn't contain anything from the last phone number, and if the program still execute normally.	A.RB.6.S.2.X.3. Append phone number to full phone number list exception case

EXAMPEL Phonebook: Verification

- Verification 1 of white-box

Actual outcome and pass-fail conclusion	Expected outcome and pass-fail criteria	Requirement
VR.1. Pass	XP.1.	A.RR.1.1.
VR.1. Pass	XP.1.	A.RR.2.1.
VR.1. Pass	XP.1.	A.RR.4.1.
VR.1. Pass	XP.1.	A.RR.3.1.
VR.1. Pass	XP.1.	A.RR.5.1.
VR.1. Pass	XP.1.	A.RR.5.2.
VR.1.1. Fail, At equal string length (C.2. and C.5.) or at one character more (C.3. and C.4.), the debugger gave Assertion failed, see Figure 12-34 below.	XP.1. Pass, if oversized person data and phone numbers get truncated at their max length and if the program continues to execute normally and if these data keep after termination and restart.	A.RR.5.4. MAX_PERS_LEN = 128 A.RR.5.5. MAX_NUMB_LEN = 32

- Verification 2 of black-box

Actual outcome and pass-fail conclusion	Expected outcome and pass-fail criteria	Requirement
VR.2. Pass	XP.1.	A.RB.6.S.1.U.1.
VR.2. Pass	XP.1.	A.RB.6.S.1.X.2.
VR.2. Pass	XP.1.	A.RB.6.S.1.X.3.
VR.2.1. Fail, For all cases C.a. to C.r. last contact was not read. For example, if unexpected file end occur within first contact, then the number of contacts are displayed as -1.	XP.1. Pass, if a warning message is displayed and thereafter if a welcome message is displayed, and if all intact contacts are read.	A.RB.6.S.1.X.4. Found file unexpected end exception case
VR.2. Pass	XP.1.	A.RB.6.S.2.U.1.
VR.2. Pass	XP.1.	A.RB.6.S.2.X.2.
VR.2. Pass	XP.1.	A.RB.6.S.2.X.3.
VR.2. Pass	XP.1.	A.RB.6.S.3.U.1.
VR.2. Pass	XP.1.	A.RB.6.S.3.U.2.
VR.2. Pass	XP.1.	A.RB.6.S.4.U.1.
VR.2. Pass	XP.1.	A.RB.6.S.4.X.2.
VR.2.2. Fail, The last contact is deleted even if a contact number one too high is used.	XP.1. Pass, if the program display an error message that the used contact number is out of existing range.	A.RB.6.S.4.X.3. Delete contact not existing exception case
VR.2. Pass	XP.1.	A.RB.6.S.5.U.1.
VR.2.3. Fail Displayed failure message is correct, but despite the phone-book program terminates	XP.1. Pass, if the program display an error message that the phone-book to be saved can not be created.	A.RB.6.S.5.X.2. File not created exception case
VR.2.4. Fail, Despite not anything is saved, this message is not displayed	XP.1. Pass, if the program display an error message that not anything are saved.	A.RB.6.S.5.X.3. File not written exception
VR.2. Pass	XP.1.	A.RB.6.S.5.X.4.

- Locate failure
- Make failure elimination plan
- Correct
- Re-verify

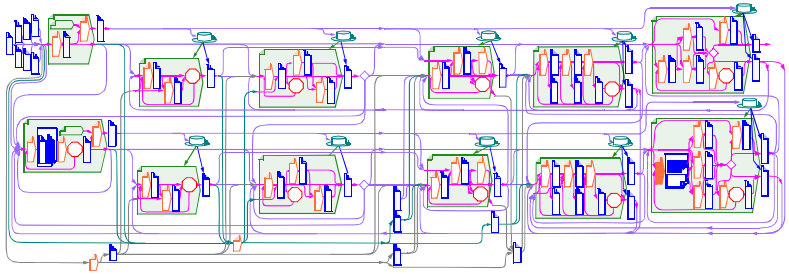
Primary failure	Failure elimination plan	Actual outcome and pass-fail criteria	Expected outcome and pass-fail criteria	Restrictions and use case requirement
FER.1.1. The safe C instruction gets_s take a string length size parameter. If input exceeds, the program stops.	FER.1.1.a.Input strings are long and rarely overloaded. FER.1.1.b. But a professional program should not allow runtime crashes. FER.1.1.c. Catch the failure before the program is stopped and truncate the string at max size FER.1.1.d. Reverify	VR.1.1. Fail, At equal string length (C.2. and C.5.) or at one character more (C.3. and C.4.), the debugger gave Assertion failed, see Figure 12-34 below.	XP.1. Pass, if oversized person data and phone numbers get truncated at their max length and if the program continues to execute normally and if these data keep after termination and restart.	A.RR.5.4. MAX_PERS_LEN = 128 A.RR.5.5. MAX_NUMB_LEN = 32
FER.1.2. A simple logic failure when calculating number of intact contacts	FER.1.2.a.Find the failure by the debugger FER.1.2.b. Correct the failure FER.1.2.c. Reverify	VR.2.1. Fail, For all cases C.a. to C.r. last contact was not read. For example, if unexpected file end occur within first contact, then the number of contacts are displayed as -1.	XP.1. Pass, if a warning message is displayed and thereafter if a welcome message is displayed, and if all intact contacts are read.	A.RB.6.S.1.X.4. Found file unexpected end exception case
FER.1.3. A simple logic failure when comparing with number of last contact	FER.1.3.a.Find the failure by the debugger FER.1.3.b. Correct the failure FER.1.3.c. Reverify	VR.2.2. Fail, The last contact is deleted even if a contact number one too high is used.	XP.1. Pass, if the program display an error message that the used contact number is out of existing range.	A.RB.6.S.4.X.3. Delete contact not existing exception case
FER.1.4. The file is ordered to close, despite it not created	FER.1.4.a.Change the file closing instruction to after it first has been opened. FER.1.4.b. Reverify	VR.2.3. Fail Displayed failure message is correct, but despite the phone-book program terminates	XP.1. Pass, if the program display an error message that the phone-book to be saved can not be created.	A.RB.6.S.5.X.2. File not created exception case
FER.1.5. When writing to the file, the first lines are buffered and impossibility to write not immediately detected.	FER.1.5.a.Flush the buffer after the first line is written, then the impossibility to write is discovered. FER.1.5.b. Reverify	VR.2.4. Fail, Despite not anything is saved, this message is not displayed	XP.1. Pass, if the program display an error message that not anything are saved.	A.RB.6.S.5.X.3. File not written exception

EXAMPLE Phonebook: Failure elimination and manual

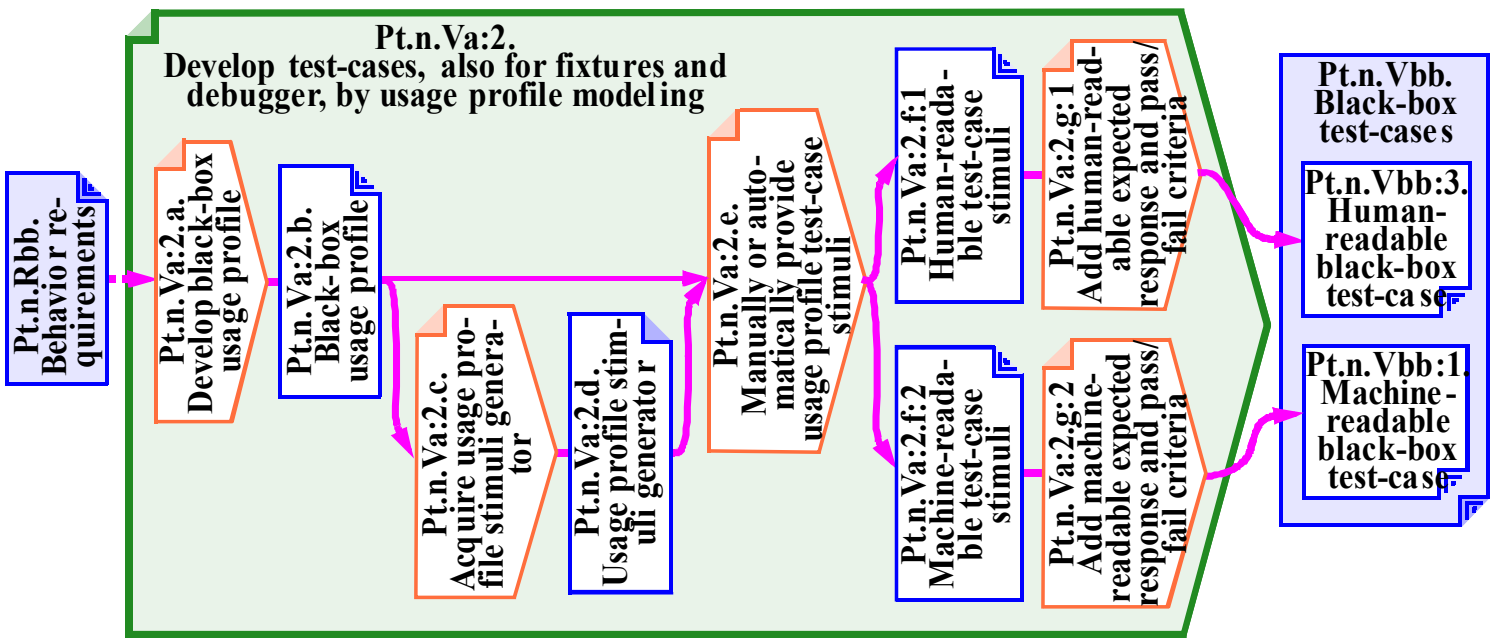
- Plan manual chapters from each developed white-box

Manual table of content	Requirement / design
MA.1. Starting up the program, including exception handling	A.RB.6.S.1. Initiate phone-book scenario
MA.2. How to append a contact to the phone-book, including exception handling	A.RB.6.S.2. Append contact to phone-book scenario
MA.3. How to list all contacts, including exception handling	A.RB.6.S.3. List phone-book contacts scenario
MA.4. How to delete a contact, including exception handling	A.RB.6.S.4. Delete phone-book contact scenario
MA.5. Terminating the program, including exception handling	A.RB.6.S.5. Terminate phone-book scenario
MA.6. Implemented phone-book size limitations and proposed way of extension	A.RR.5.2. MAX_CONTACTS = 128 A.RR.5.3. MAX_NUMB_PERS = 16
MA.7. Implemented string length limitations	A.RR.5.4. MAX_PERS_LEN = 128 A.RR.5.4. MAX_PERS_LEN = 128
MA.8. Architecture rationals and proposed way of extension	A.Phone-book logical architecture (Figure 9-63 on page 412)
MA.9. Object oriented rationals and proposed way of extension	A.Phone-book updated logical architecture (Figure 10-55 on page 534)

Pt.n.Va:2. Develop test cases, also for fixtures and debugger, by using usage profile modelling

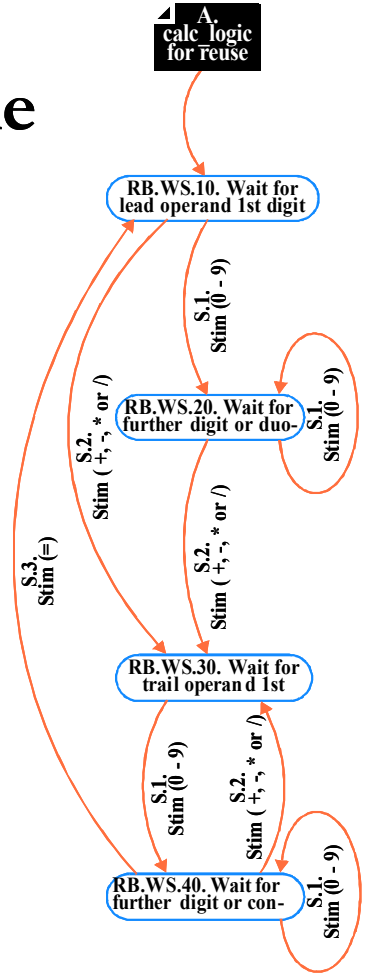


- Activity Pt.n.Va. extended to schedule Pt.n.Va:2.
- Develop black-box usage profiles
- Generate stimuli from usage profile
- Add expected response and pass/fail criteria



EXAMPLE calc_logic: Develop black-box usage profile

- Begin with use case scenario and transform to state machine
- Assign probability value to all state transitions, that reflect how much the user operate each transition
- Many user profiles may be assumed
  - Basic user
  - Advanced user
  - Stressed user



Use case with input	Calc_logic usage profiles, estimations in percent			Next state
	Basic usage profile	Advanced usage profile	Stressed usage profile	
	RB.WS.10. Wait for lead operand 1st digit			
S.1. Stim (0 - 9)	10 % per digit	8 % per digit	2 % per digit	RB.WS.20.
S.2. Stim (+, -, * or /)		5 % per operator	10 % per operator	RB.WS.30.
S.3. Stim (=)			40 %	?.
Sum	100 %	100 %	100 %	
	RB.WS.20. Wait for further digit or duo-operator			
S.1. Stim (0 - 9)	5 % per digit	8 % per digit	2 % per digit	RB.WS.20.
S.2. Press (+, -, *, /)	30 % 10 % 5 % 5 %	5 % 5 % 5 % 5 %	20 % 20 % 20 %	RB.WS.30.
S.3. Stim (=)			20 %	?.
Sum	100 %	100 %	100 %	
	RB.WS.30. Wait for trail operand 1st digit			
S.1. Stim (0 - 9)	10 % per digit	10 % per digit	4 % per digit	RB.WS.40.
S.2. Stim (+, -, * or /)			10 % per operator	?.
S.3. Stim (=)			20 %	?.
Sum	100 %	100 %	100 %	
	RB.WS.40. Wait for further digit or con-clusion operator			
S.1. Stim (0 - 9)	5 % per digit	7 % per digit	2 % per digit	RB.WS.40.
S.3. Stim (=)	30 %	10 %	40 %	RB.WS.10.
S.2. Press (+ -, * or /)	20 %	8 % 4 % per operator	22 % 6 % per operator	RB.WS.30.
Sum	100 %	100 %	100 %	

EXAMPLE calc_logic: Generate usage profile machine readable stimuli (comma separated)

- Basic usage
  - Mostly addition and subtraction
- Advanced usage
  - Also multiplication and division
- Stressed usage
  - Strangely few numbers

```
8, 7, +, 4, =, 5, +, 5, 9, 7, +, 2, 8, 7, =, 6, +, 5, =, 3, +, 8, +, 8, =, 7, /, 2, 0,
8, =, 9, 3, -, 2, +, 1, 0, 7, 3, 2, 6, 4, 8, =, 8, 2, +, 8, 9, 4, =, 6, -, 5, +, 7, =,
0, +, 3, =, 9, 2, 1, 0, 0, 8, +, 4, =, 6, 6, *, 7, =, 5, +, 9, 7, 8, 3, 8, 6, 7, 9, 9,
9, 5, 4, 2, =, 2, /, 6, =, 4, 2, 4, 6, 3, 0, +, 7, 5, 2, 4, =, 7, 2, 7, 8, +, 0, +, 1,
+, 4, =, 9, 9, 4, 4, -, 4, 5, +, 1, 5, 7, 2, 9, =, 2, 7, 1, 5, 4, *, 7, =, 1, 1, +, 2,
=, 3, *, 1, 7, 0, =, 3, 0, +, 8, 4, 1, 6, 3, =, 2, -, 0, 8, =, 2, +, 1, 4, =, 4, 9, 7,
1, 6, +, 2, 8, 6, =, 7, 8, +, 5, =, 4, 8, +, 7, +, 7, 9, 4, =, 3, 4, 9, 7, +, 0, +, 6,
=, 8, +, 3, =, 1, /, 6, +, 9, 8, +, 3, 5, =, 2, /, 6, 8, =, 2, +, 6, =, 9, -, 1, 9, =,
3, +, 8, 5, 0, =, 4, 5, 1, /, 5, 7, 6, 9, 9, +, 2, 0, =, 8, +, 2, 8, =, 0, /, 2, =, 6,
+, 4, +, 0, =, 0, 1, 6, +, 3, +, 6, 6, 7, +, 8, 7, 0, 1, =, 8, -, 1, 0, =, 4, 0, -, 3,
+, 5, 2, 9, 8, +, 5, =, 8, +, 9, =, 6, 9, 4, 5, 6, 6, 1, +, 5, 5, =, 3, +, 6, 2, 5, +,
8, +, 4, 2, 4, =, 7, +, 6, +, 0, =, 1, 3, +, 7, 7, 8, 4, =, 5, +, 7, 5, =, 2, 0, 3, +,
3, =, 5, 4, *, 6, 6, =, 1, 2, +, 9, 6, 1, 3, 5, 8, 7, =, 0, 3, +, 1, 2, 5, 7, 2, 0, =,
3, /, 5, +, 1, 6, =, 2, 2, +, 8, +, 8, +, 6, 0, 5, 7, 8, +, 3, 4, =
```

```
5, 7, +, 6, /, 9, 6, 6, 1, 6, +, 8, 0, -, 3, 3, 6, 7, 3, 8, 0, 9, 9, =, /, 0, 8, 9, 9,
9, +, 9, /, 5, 3, 0, 6, 7, 8, 6, +, 5, 8, 0, 3, 6, 2, +, 4, 4, 7, +, 4, +, 8, 0, 5, 4,
2, =, 8, 3, -, 6, 5, 5, =, -, 9, 8, 4, *, 7, 4, *, 3, =, +, 5, 1, 5, 4, 8, 8, 3, 5, 6,
9, 2, 2, 7, 3, 1, 3, *, 7, =, 1, +, 2, 4, 4, 1, 0, 7, 4, 9, -, 5, 6, 1, 9, 2, -, 0, 8,
-, 8, 4, -, 9, 8, 4, 3, +, 1, 1, 3, -, 4, +, 4, 1, 2, 2, +, 4, 7, 5, 0, 0, 7, 2, *, 3,
4, 2, +, 7, 2, 3, 9, 1, 1, +, 1, 2, 6, 8, 0, 8, +, 9, 0, 0, 2, 6, 8, 2, 4, 0, -, 2, =,
+, 1, +, 3, 0, 6, 4, 2, 0, 9, 3, 3, 8, 1, 7, 8, 6, 5, *, 1, 9, 8, 3, *, 1, 6, 2, 8, 1,
=, 3, 6, +, 5, 7, 6, 6, *, 2, 3, 4, +, 0, 4, 6, 0, 6, 5, 3, /, 2, 4, 3, 4, 3, +, 2, 0,
+, 6, =, 0, 4, 2, 8, 8, 5, 2, 2, -, 4, -, 8, 7, 9, =, 9, 4, 4, 2, +, 6, 6, 1, 4, 6, 1,
9, =, 4, *, 1, =, 2, 3, +, 9, 5, +, 6, =, 4, 0, 1, 9, 0, 6, 1, -, 5, 6, 3, 7, *, 1, 7,
4, =, 3, 4, 4, 3, 7, +, 9, 7, 8, 5, 5, +, 3, 8, 2, 3, 7, 7, =, 3, 6, 7, -, 9, 2, 7, 8,
=, 4, *, 6, 8, 4, 4, 5, 6, 9, 9, *, 1, 6, 3, 8, =, 7, 4, 6, 5, 9, 9, 1, 0, 2, 9, 5, 0,
9, 6, 0, 2, 0, 3, 8, 7, *, 4, 5, 0, 7, *, 3, 0, 3, =, 6, 2, 8, -, 1, 3, 3, +, 1, 5, 5,
7, 0, 9, 5, 3, 3, 4, 9, 0, *, 6, 8, 7, =
```

```
2, -, =, -, *, -, *, 7, =, 8, =, *, *, +, 7, =, 0, =, /, =, =, =, =, =, 8, -, +, 1, +,
8, =, 4, 2, -, *, /, /, +, 7, =, *, =, =, +, *, =, =, 4, *, =, 9, =, =, =, *, 0, 0, /,
=, =, 2, -, =, =, =, =, =, =, /, /, -, =, =, /, +, =, =, =, *, =, +, =, =, /, =, 5, *,
*, =, =, =, -, =, 8, =, 2, *, 5, =, /, *, =, 4, *, =, 0, /, =, =, +, -, +, =, 5, /, 0,
+, 1, =, /, 8, *, -, +, 7, 2, 7, 6, =, =, =, 2, -, 6, =, =, +, *, +, =, =, =, *, =, =,
*, -, 5, -, 6, =, -, -, 4, *, 0, +, -, *, +, =, 0, /, =, -, /, *, 1, -, *, 1, =, =, =,
/, +, =, 5, -, 1, +, 5, +, 8, =, =, =, 1, 5, -, =, =, =, /, =, 2, 4, /, =, =, 2, -, +,
*, 3, =, =, =, =, /, =, =, /, /, *, -, =, *, /, =, 6, -, 3, =, *, =, 7, *, 1, =, +, 5,
=, 1, -, *, *, 1, /, -, +, +, =, =, /, *, /, 4, =, =, /, =, 0, =, =, =, =, *, -, 0, 4,
8, =, 5, *, /, 6, +, *, 4, 3, =, =, =, =, =, +, /, =, /, =, +, =, =, =, 0, -, *, 7, 8,
7, 7, /, *, -, 3, *, 4, +, 8, /, =, 3, /, =, /, 8, +, =, /, 6, +, -, =, *, -, -, -, 2,
=, -, -, =, =, =, =, =, =, 3, 0, *, +, -, *, 8, =, -, =, =, =, =, /, +, *, =, 2, *, *,
/, 9, /, =, =, =, 7, =, *, +, =, -, 3, -, *, =, =, +, /, +, 5, =, *, 6, =, =, 0, 6, -,
=, -, 5, =, =, -, =, +, -, 6, /, 9, =, =, +, =, 6, *, *, 6, =, -, =
```

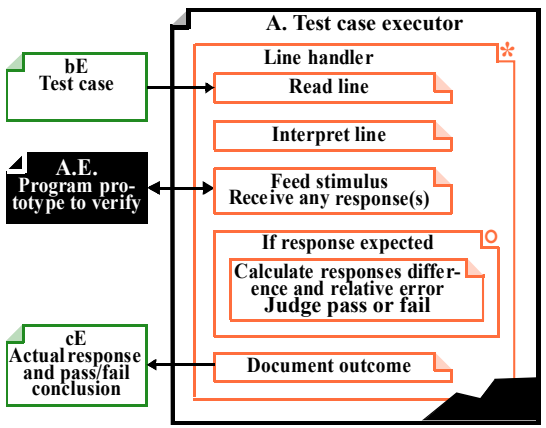
EXAMPLE calc_logic: Create test case by adding expected responses and pass/failure criteria

- Generated expected responses and pass/failure criteria
- In this example, after any number no response is expected
- A semicolon indicates an expectation of response
- Since division easily result in inexact quotients, also a minimum accuracy is specified for passing

```
8
7
+
4
=; 91 0
5
+
5
9
7
+; 602 0
2
8
7
=; 889 0
6
+
5
=; 11 0
3
+
8
+; 11 0
8
=; 19 0
7
/
2
0
8
=; 0.033653846153846153846153846153846 1e-16
9
3
-
2
+; 91 0
1
0
7
3
2
6
4
8
=; 10732739 0
```

EXAMPLE calc_logic:
Apply automatic test case executor to wrap calc_logic

- The executor read from the from the test case and feed program prototype
- The executor listen for responses from the program prototype
- Received responses are compared with the expected response
- The difference must be lower than specified allowed accuracy
- Observe that the test case executor must calculate with better accuracy than is demanded from the product prototype



EXAMPLE calc_logic:
Document outcome from the test case executor

- Example of automatic verification report
- When an failure has been detected, it may be difficult to trace to when it happened
- Run the executor and product prototype in a debugger to set halt conditions on failures

```
 :
 :
178 Stimulus =; Response 10852951649; Expected 10852951649;
  Rel. error  0.0; Max allowed 0; PASS
179 Stimulus +
180 Stimulus 1; Response 1
181 Stimulus +; Response 10852951650; Expected 10852951650;
  Rel. error  0.0; Max allowed 0; PASS
182 Stimulus 3; Response 3
183 Stimulus 0; Response 30
184 Stimulus 6; Response 306
185 Stimulus 4; Response 3064
186 Stimulus 2; Response 30642
187 Stimulus 0; Response 306420
188 Stimulus 9; Response 3064209
189 Stimulus 3; Response 30642093
190 Stimulus 3; Response 306420933
191 Stimulus 8; Response 3064209338
192 Stimulus 1; Response 30642093381
193 Stimulus 7; Response 306420933817
194 Stimulus 8; Response 3064209338178
195 Stimulus 6; Response 30642093381786
196 Stimulus 5; Response 306420933817865
197 Stimulus *; Response 306431786769515; Expected 306431786769515;
  Rel. error  0.0; Max allowed 0; PASS
198 Stimulus 1; Response 1
199 Stimulus 9; Response 19
200 Stimulus 8; Response 198
201 Stimulus 3; Response 1983
202 Stimulus *; Response 6.076542331639483e+017; Expected
607654233163948245;
  Rel. error  0.00000000000000009; Max allowed 0; FAIL
203 Stimulus 1; Response 1
204 Stimulus 6; Response 16
205 Stimulus 2; Response 162
206 Stimulus 8; Response 1628
207 Stimulus 1; Response 16281
208 Stimulus =; Response 9.893218570142241e+021; Expected
9893218570142241376845;
  Rel. error  0.00000000000000003; Max allowed 0; FAIL
 :
 :
```