Lero THE IRISH SOFTWARE RESEARCH CENTRE

Evolving Critical Systems

Prof. Mike Hinchey

University of Limerick · Trinity College Dublin · DCU · UCD Dublin · OÉ Gaillimh NUI Galway · Dundalk Institute of Technology · UCC University College Cork, Ireland · Maynooth University National University of Ireland Maynooth

ACM Highlights

- Learning Center tools for professional development: http://learning.acm.org
  - 4,900+ trusted technical books and videos from O'Reilly, Morgan Kaufmann, etc.
  - 1,400+ courses, virtual labs, test preps, live mentoring for software professionals covering programming, data management, cybersecurity, networking, project management, more
  - 30,000+ task-based short videos for "just-in-time" learning
  - Training toward top vendor certifications (CEH, Cisco, CISSP, CompTIA, ITIL, PMI, etc.)
  - Learning Webinars from thought leaders and top practitioner
  - Podcast interviews with innovators, entrepreneurs, and award winners

- Popular publications:
  - Flagship *Communications of the ACM (CACM)* magazine: http://cacm.acm.org/
  - *ACM Queue* magazine for practitioners: http://queue.acm.org/

- ACM Digital Library, the world's most comprehensive database of computing literature: http://dl.acm.org.

- International conferences that draw leading experts on a broad spectrum of computing topics: http://www.acm.org/conferences.

- Prestigious awards, including the ACM A.M. Turing and Infosys: http://awards.acm.org

- And much more… http://www.acm.org.

"Housekeeping"

Twitter: #ACMLearning

- Welcome to today's ACM Learning Webinar, "Evolving Critical Systems" by Mike Hinchey. The presentation starts at the top of the hour and lasts 60 minutes. Slides will advance automatically throughout the event. You can resize the slide area as well as other windows by dragging the bottom right corner of the slide window, as well as move them around the screen. On the bottom panel you'll find a number of widgets, including Twitter, Sharing, and Wikipedia apps.

- If you are experiencing any problems/issues, refresh your console by pressing the F5 key on your keyboard in Windows, Command + R if on a Mac, or refresh your browser if you're on a mobile device; or close and re-launch the presentation. You can also view the Webcast Help Guide, by clicking on the "Help" widget in the bottom dock.

- To control volume, adjust the master volume on your computer. If the volume is still too low, use headphones.

- If you think of a question during the presentation, please type it into the Q&A box and click on the submit button. You do not need to wait until the end of the presentation to begin submitting questions.

- At the end of the presentation, you'll see a survey open in your browser. Please take a minute to fill it out to help us improve your next webinar experience.

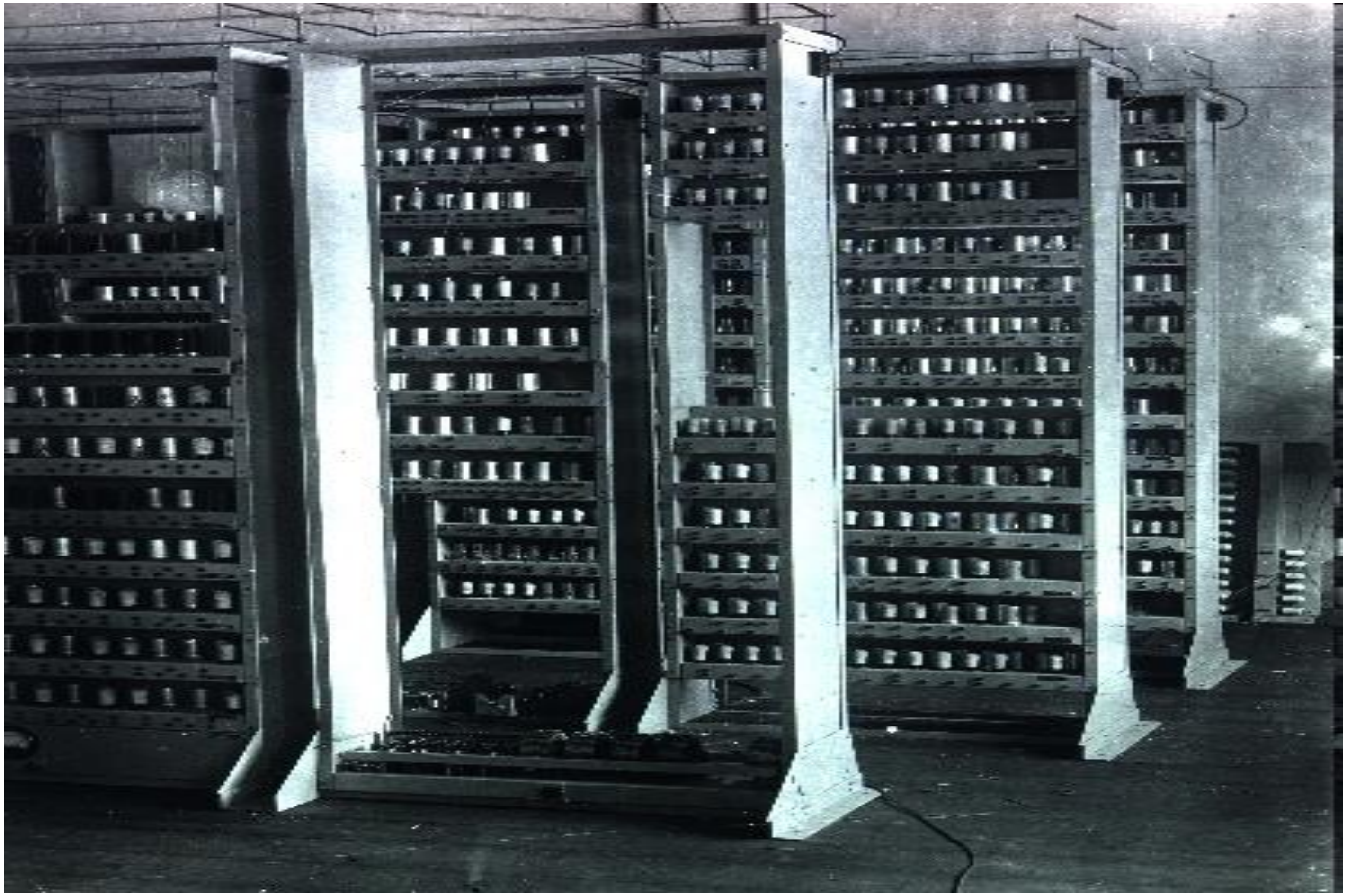
- You can download a copy of these slides by clicking on the Resources widget in the bottom dock.

- This session is being recorded and will be archived for on-demand viewing in the next 1-2 days. You will receive an automatic email notification when it is available, and check http://learning.acm.org/ in a few days for updates. And check out http://learning.acm.org/webinar for archived recordings of past webcasts.

Talk Back

- Use Twitter widget to Tweet your favorite quotes from today's presentation with hashtag #ACMLearning
- Submit questions and comments via Twitter to @acmeducation – we're reading them!
- Use the sharing widget in the bottom panel to share this presentation with friends and colleagues.

66 Years Ago …



EDSAC

- 650 instructions per second.
- 1024 17-bit words of memory in mercury ultrasonic delay lines.
- Paper tape input and teleprinter output at 6 2/3 characters per second.
- 3000 valves, 12 kW power consumption, occupied a room 5m by 4m.
- "Operating system" occupied 31 words of read-only memory.
- Early use to solve problems in meteorology, genetics and X-ray crystallography.

Difference Engine

Motivation

Errata, detected in Taylor's Logarithms*. London: 4to, 1972 [sic]*

…

Kk Co-sine of 14.18.3 – 3398 – 3298

Nautical Almanac (1832)

…

In the list of ERRATA detected in Taylor's *Logarithms*, for cos. 4 18' 3'' read cos. 14 18'2''.

Nautical Almanac (1833)

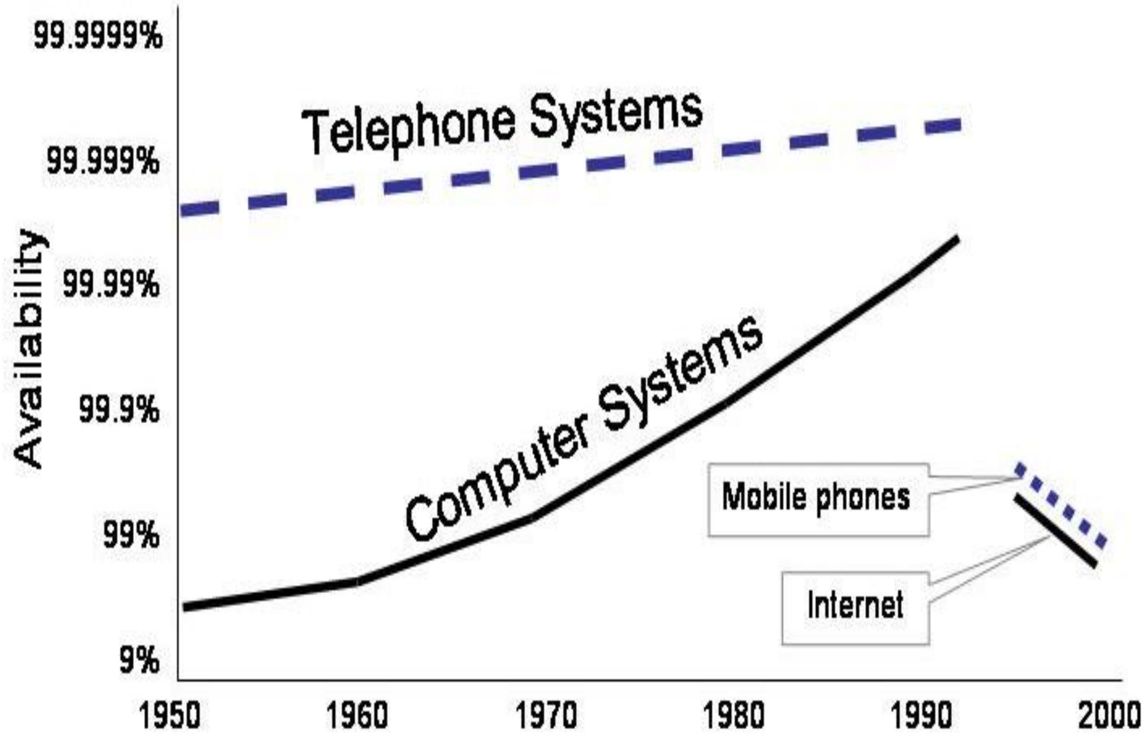
ERRATUM of the ERRATUM of the ERRATA of TAYLOR'S *Logarithms.* For cos. 4 18'3'', *read* 14 18' 3''.

Nautical Almanac (1836)

First Programmer

Augusta Ada King, Countess of Lovelace

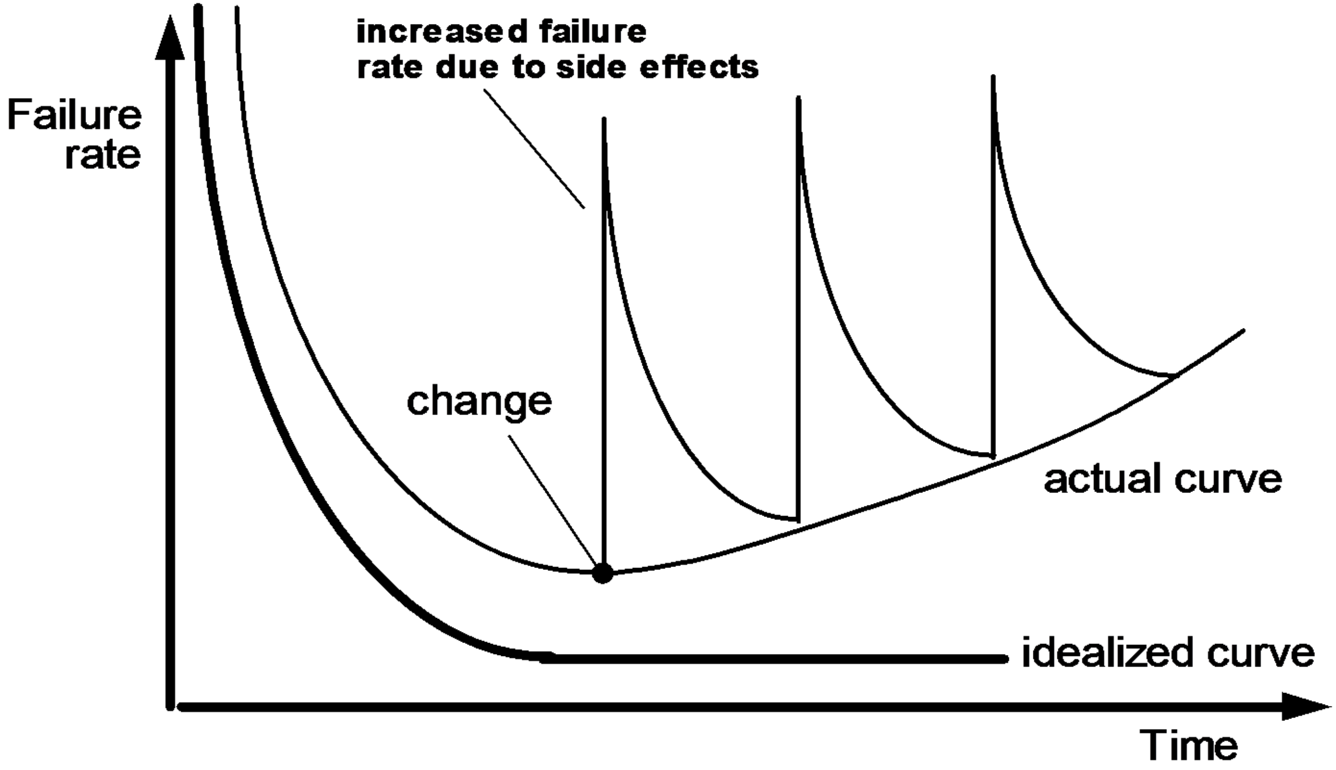
Software Lags behind Hardware



Software vs. Hardware

- Pervasive yet non-obvious;
- Abstract as opposed to “concrete”;
- Perceived to be “easy to change”;
- Easy-to-change means often changed;
- Not visibly deteriorating.

Wear versus Deterioration



Major Software Failures

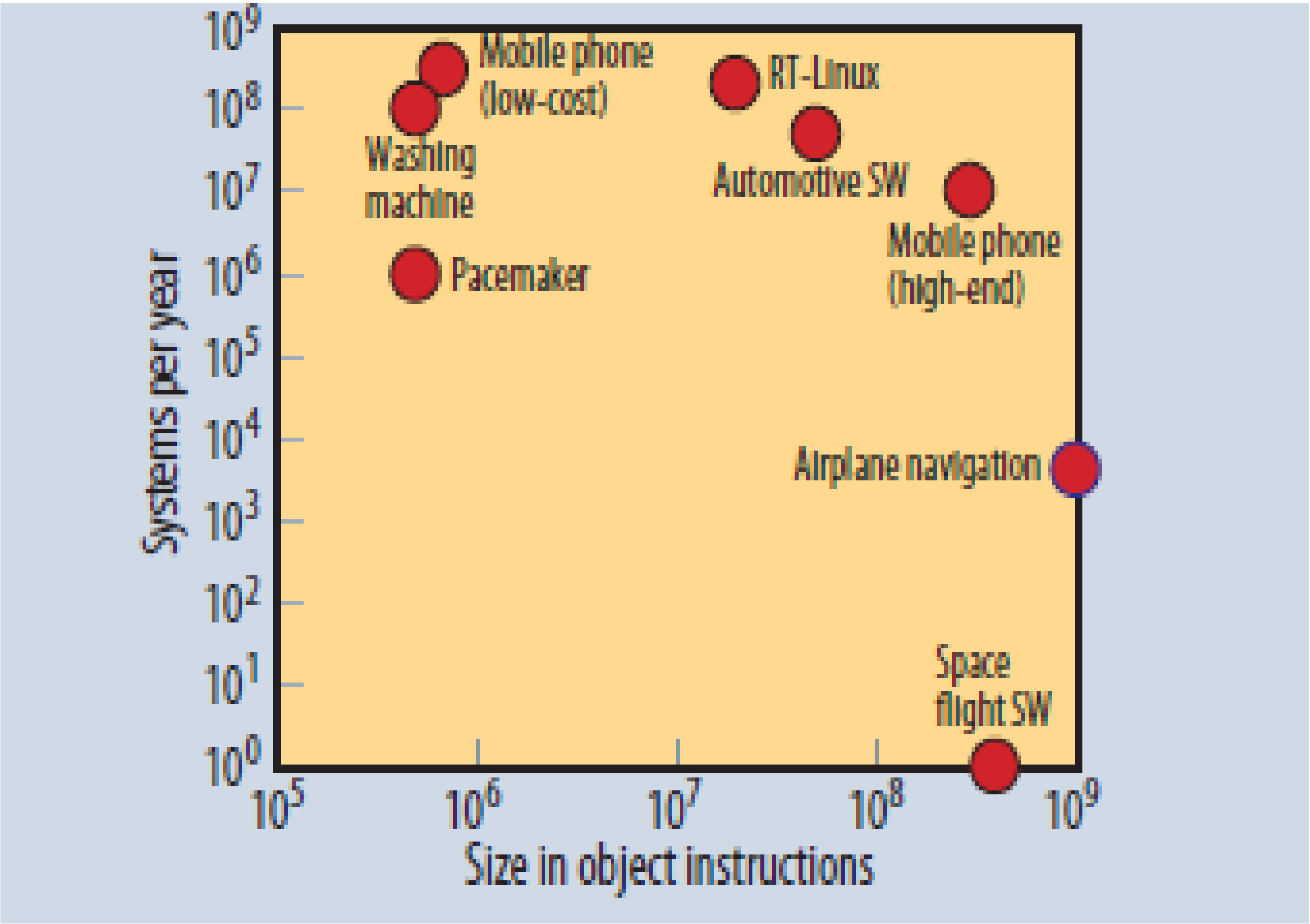
- Therac 25
- ARIANE 5
- Mars Polar Lander
- … *and many many more!*

Problem

The problem is Complexity.

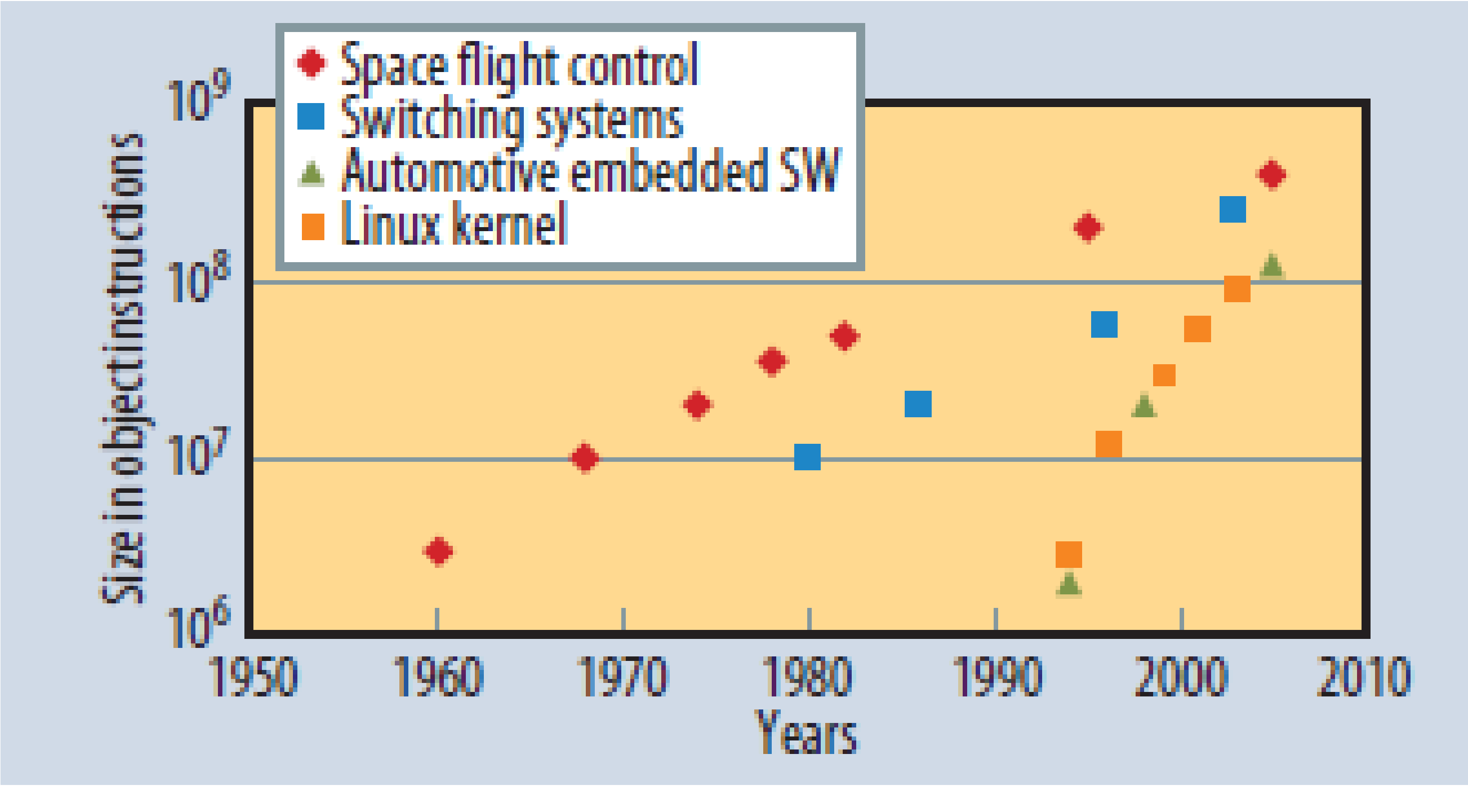
Bill Gates

Size of Modern Applications



Source: Ebert & Jones, *Computer*, April 2009

Increasing Size



Source: Ebert & Jones, *Computer*, April 2009

Challenges for Software Engineering

- Increases in demand for greater, more complex functionality;
- Stricter (required and desirable) constraints on performance and reaction times;
- Attempts to increase productivity and reduce costs while constantly pushing requirements to the limit;
- Requirement of regular change and evolving systems.

Evolution

Any intelligent fool can make things bigger and more complex …

It takes a touch of genius and a lot of courage to move in the opposite direction.

Albert Einstein

Evolving Systems

Software is not static

Software is supposed to change… otherwise it would be in the hardware!

At r… ms need t…

- to react to cha… environment;
- to meet necessary constraints on the system that were not previously satisfied and possibly not previously known;
- to protect the system from external threats.

Legacy systems are those that have evolved over longer timeframes, due to:

- …arate systems being … together;
- …oftware … tec…
- new user re…
- new regulatory co… e requirements.

Critical Systems

- Systems where failure or malfunction will lead to significant negative consequences.
- Strict requirements for security and safety to protect the user or others.
- Critical to the organization's mission, product base, profitability or competitive advantage.

Automotive Systems

Medical Devices

Financial / Enterprise Information Systems

Current Situation

- Software is pervasive, widely used, and often invisible.
- Much legacy code, badly structured, poorly maintained.
- Many software failures, declining quality:
  - E.g., Therac 25, ARIANE 5, Mars Polar Lander, … and many more!
- Complex physical environments and diverse hardware platforms.
- Insufficient number of qualified developers and testers.
- Current techniques do not scale sufficiently and have failed to overcome 50 years of declining quality.

Evolving Critical Systems

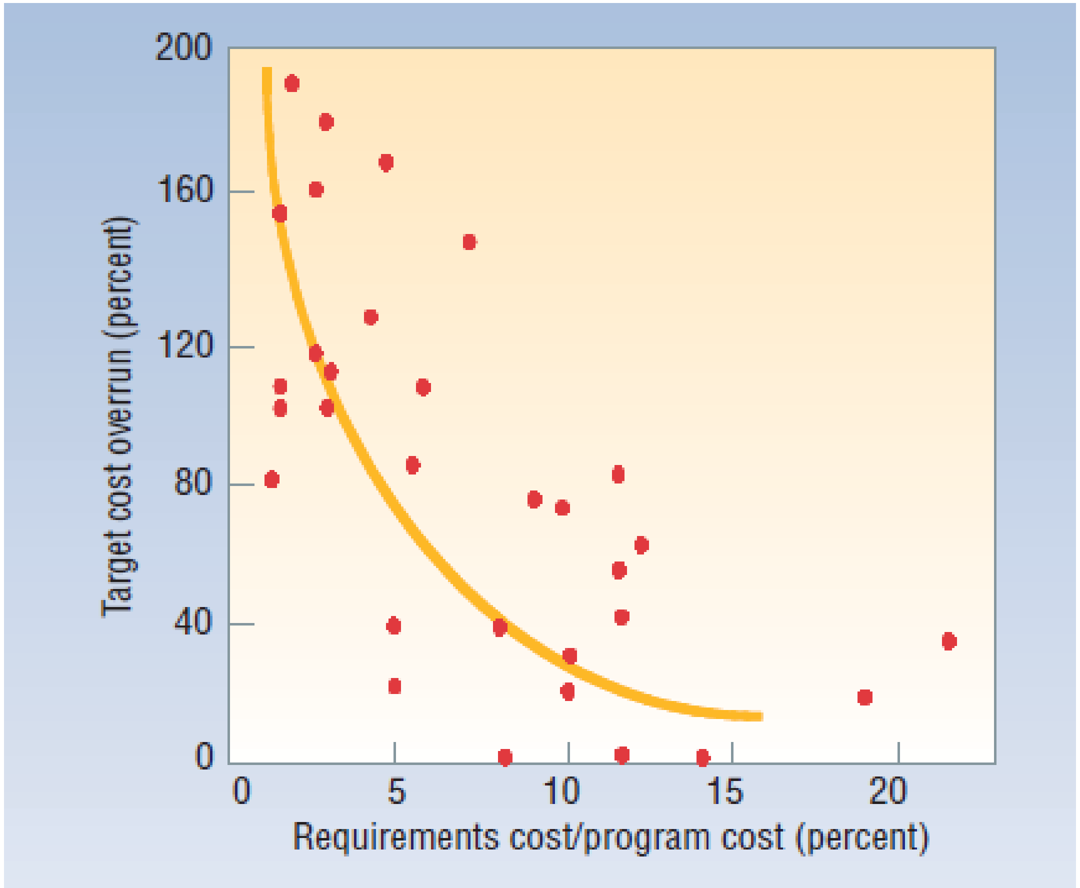
- have evolved from legacy code and legacy systems, or
- result from a combination of existing component-based systems, possibly over significant periods of time, or
- evolve as a result of a focused and intentional change in organization and architecture to exploit newer techniques believed to be beneficial;
- they require that the system adapt and evolve at run-time in order to react to changes in the environment or to meet necessary constraints on the system that were not previously satisfied and possibly not previously known.

ECS

An Evolving Critical System must be

- described in a manner that enables the developer to understand the necessary functionality of the system (*requirements engineering*), and
- which are expressed in a clear and precise way (*formal specification*),
- and yet which offers sufficient flexibility to follow the processes and practices within the organisation or necessitated by the development process (*agile methods, software processes, software process improvement*).

Requirements Effort vs. Cost Overrun



ECS

The architecture of the system must be well understood;

- the *architecture* may be the basis for future decisions on changes to be made as part of the evolution process;
- this is particularly true where the system evolves at run-time (*adaptive systems, autonomic computing, organic computing*);
- models of the system are a key component (*model driven development*), which will change over time and offer insights into potential areas of difficulty and as the basis for (possibly automated) code-generation.

ECS

An ECS must be structured

- in a way that change can be controlled and clear,
- with fixed core functionality
- and then features that may be changed, adapt, and even be deleted (*software product lines*) in order to support the necessary evolution.

ECS

Determining that quality and reliability are not impaired involves

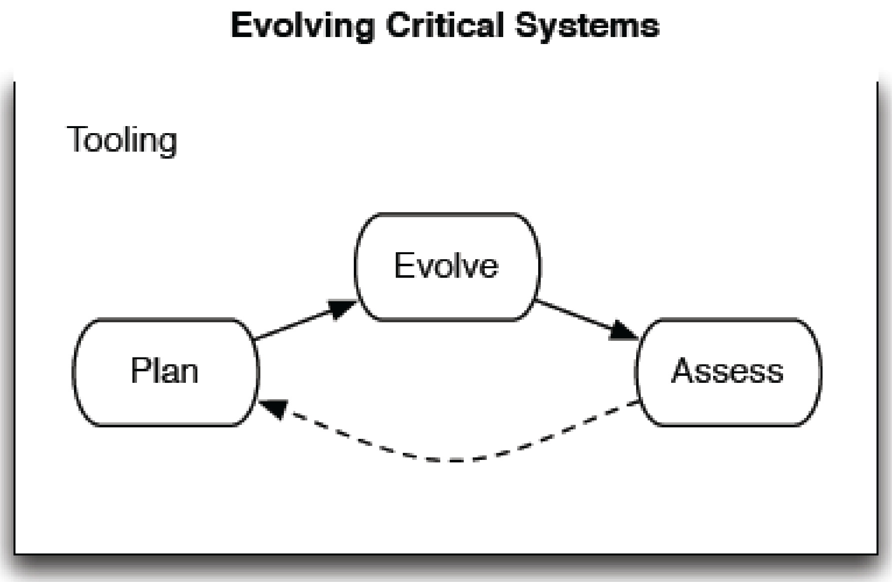
- continual overview of the development and evolutionary process (*processes and methods*, *process evaluation*);
- ensuring that policies and constraints are met (*autonomic computing, organic computing, adaptive systems*);
- collecting and recording data and evidence (*metrics, software process improvement*), and
- computation of a range of reliability measures at various points in time and the appropriate analysis thereof (*software reliability engineering*).

ECS Research Agenda

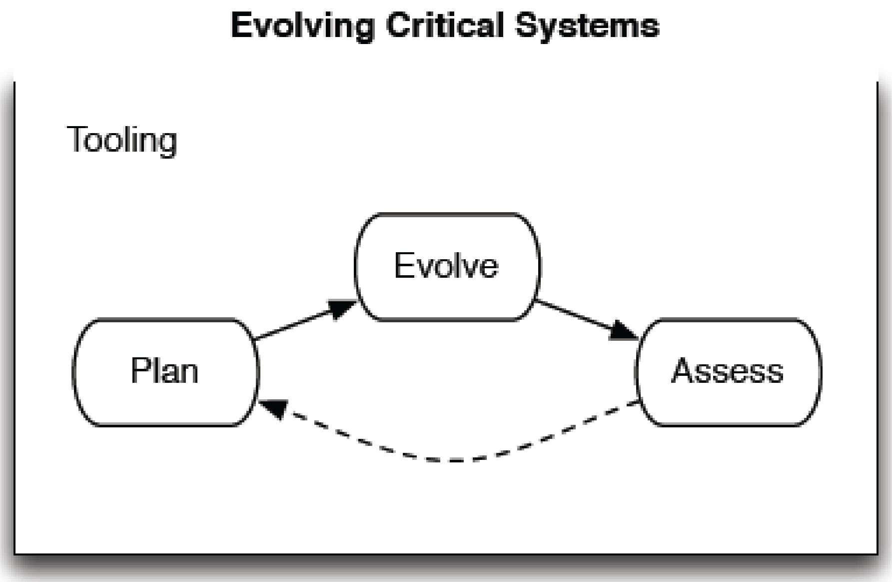
An ECS Research Agenda addresses several core research topics in the evolving critical systems field.

- The central research topic is **building software that**
- **(a) is highly reliable, and**
- **(b) retains this reliability as it evolves, *without* incurring prohibitive costs.**

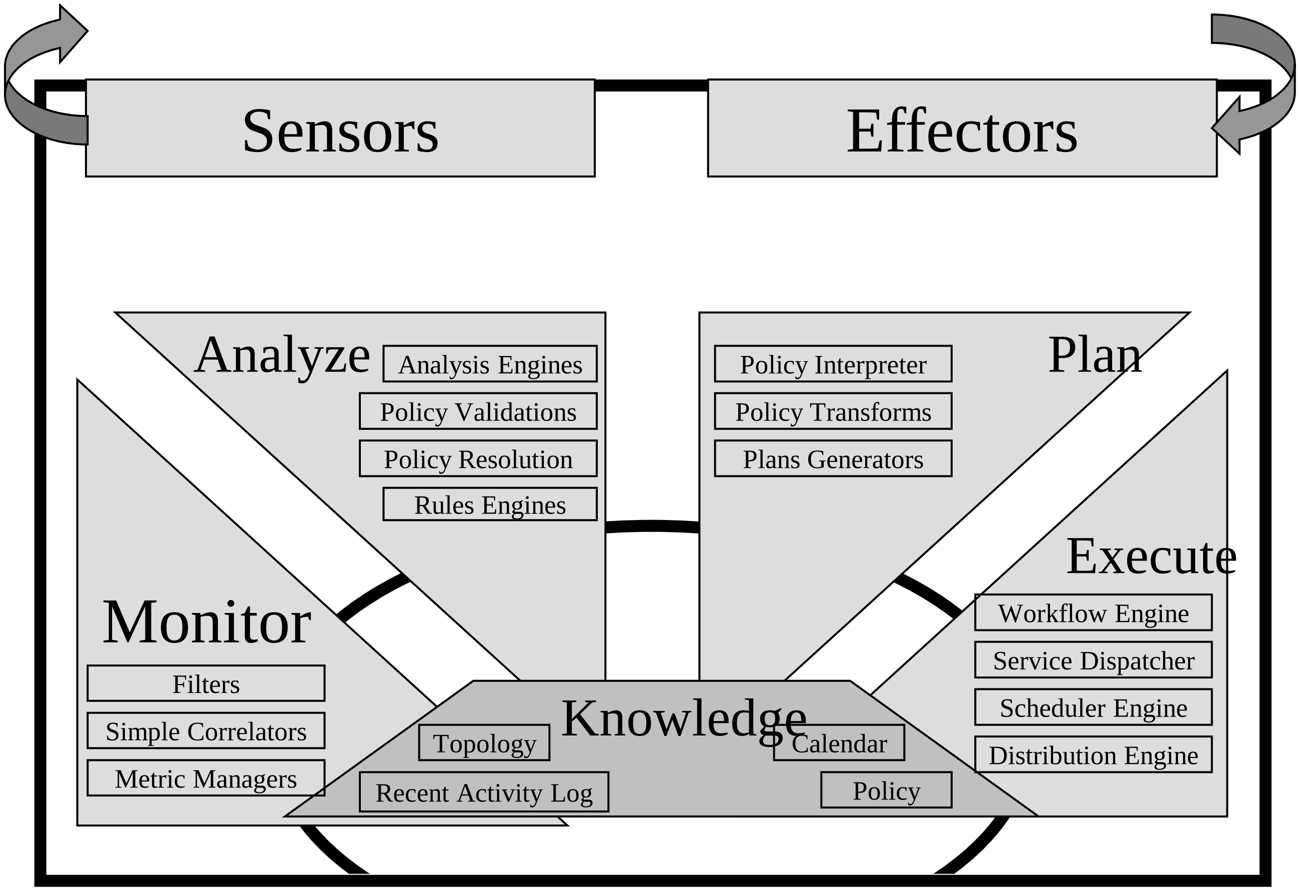
PEA+T



PEA+T



MAPE



Source: IBM, AC Blueprint 2003

Some Examples of Lero ECS Research

1. Smarter Cities
   - In conjunction with Intel Labs Europe, Dublin City Council and IBM
2. Software Product Lines
   - Use of models to gain efficiencies
3. Adaptive Security and Privacy (Cloud, smart buildings)
   - In conjunction with United Technologies and IBM
4. Parallelisation of code to optimise use of multicore hardware
   - In conjunction with Movidius and IBM

Some Examples of Lero ECS Research

5. Architectural Recovery and Preservation
   - In conjunction with several financial services companies
6. Performance Evaluation in Large Systems
   - In conjunction with IBM
7. Autonomous Space Systems
   - In conjunction with NASA and ESA and EU FP7 Project ASCENS

An ECS Scenario

Space Exploration

- Some of the most complex and expensive software applications to date.
- High Levels of Autonomy.
- Significant consequences for failure.

Swarm Technologies

- Inspired by swarms of bees and flocks of birds in nature;

- Many application areas:
  - drug discovery;
  - communication systems;
  - environmental monitoring;
  - exploration.

Swarm Technologies

Coordinated swarms of smaller spacecraft will offer:

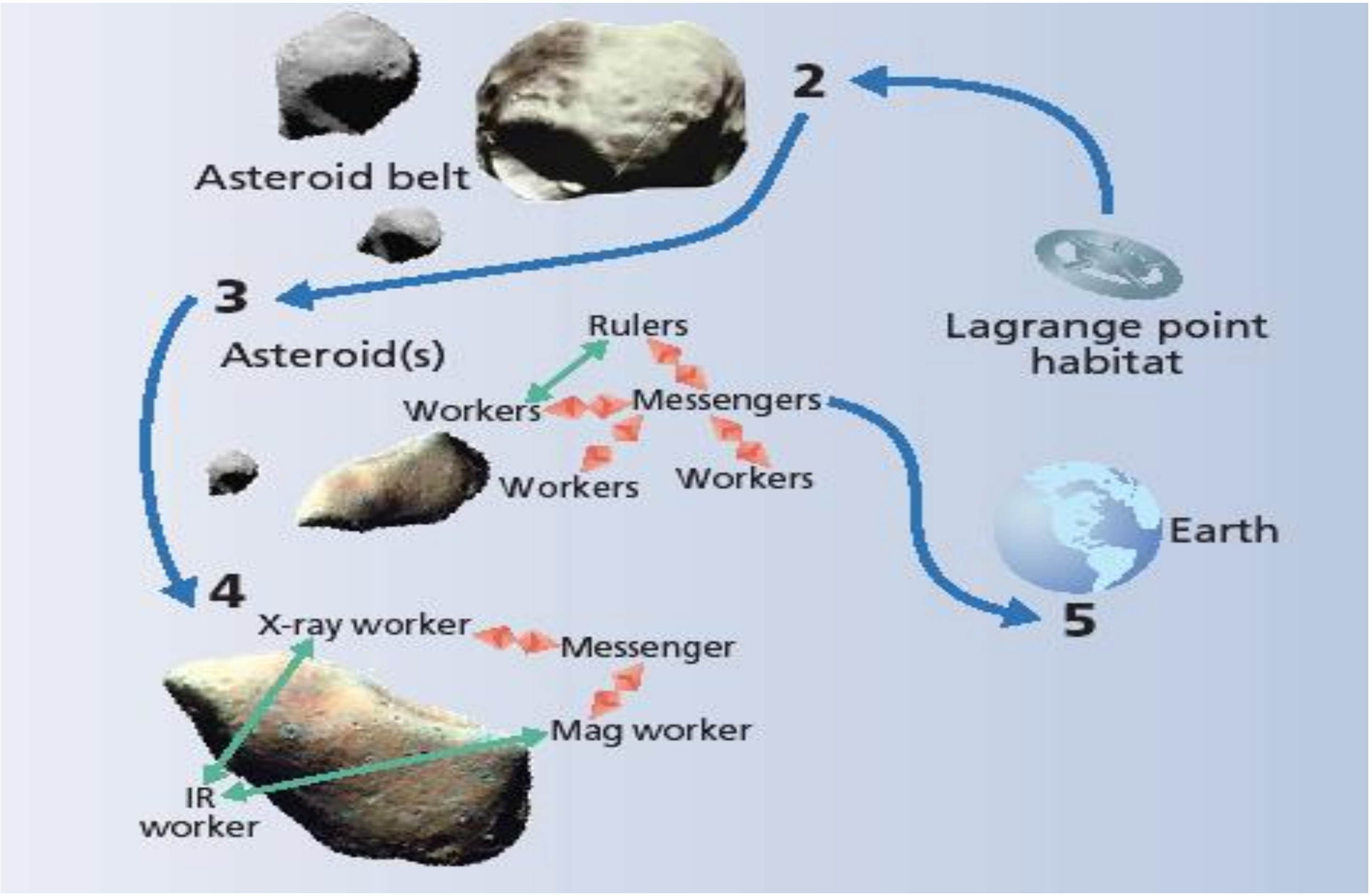
- More effective use of solar power;
- Access to areas where large craft could not go;
- Ability to perform more complex tasks;
- Greater accuracy and flexibility.

Autonomous NanoTechnology Swarm

Using swarms of “intelligent”, autonomous spacecraft to explore

1. Lunar and Martian surface (Lander Amorphous Rover Antenna, LARA)
2. Saturn’s rings (Saturn Autonomous Ring Array, SARA)
3. Asteroid belt (Prospecting Asteroid Mission, PAM)

ANTS Concept Mission - PAM



Contributions

1. Formal Methods
2. Autonomic Computing
3. Software Product Lines
4. Automatic Code Generation

Model of Formal Method

v

Swarm Formal Method Model and Outline

$$
\begin{aligned}
&LEADER\_COM_{i,conv} = leader.in?msg \rightarrow \\
&\quad \text{case } LEADER\_MESSAGE_{i,conv,msg} \\
&\qquad if \text{ sender}(msg) = LEADER \\
&\quad MESSENGER\_MESSAGE_{i,conv,msg} \\
&\qquad if \text{ sender}(msg) = MESSENGER \\
&\quad WORKER\_MESSAGE_{i,conv,msg} \\
&\qquad if \text{ sender}(msg) = WORKER \\
&\quad EARTH\_MESSAGE_{i,conv,msg} \\
&\qquad if \text{ sender}(msg) = EARTH \\
&\quad ERROR\_MESSAGE_{i,conv,msg} \\
&\qquad otherwise
\end{aligned}
$$

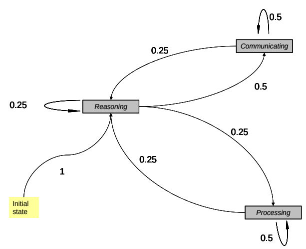
$$\Phi = \left\{ \begin{matrix} SendMessage, ReceiveMessage, \\ Reason, Process \end{matrix} \right\}$$

is a set of (partial) transition functions where each transition function maps

$$Memory \times Input \rightarrow Output \times Memory$$

$$memory' = (Goals', Model', CommsTracking)$$

[Communicating]ReasoningDeliberatve(Leader)[Reasoning]
[Reasoning] SendMessage (Leader,Worker)[Communicating]
[Processing] SendMessage (Leader, Worker) [Communicating]



$$P = \begin{pmatrix} 0 & .5 & .25 & .25 \\ 0 & .5 & .25 & .25 \\ 0 & .5 & .25 & .25 \\ 0 & 0 & .5 & .5 \end{pmatrix}$$

$$
\begin{aligned}
Communicating \equiv\ & 50\omega^2 : \text{SendMessageWorker}.Communicating \\
& + 50\omega^2 : \text{SendMessageLeader}.Communicating \\
& + 1\omega^1 : \text{SendMessageError}.Communicating \\
& + 50\omega^2 : \text{ReceiveMessageWorker}.Communicating \\
& + 50\omega^2 : \text{ReceiveMessageLeader}.Communicating \\
& + 1\omega^1 : \text{ReceiveMessageError}.Communicating \\
& + 50\omega^2 : \text{ReasoningDeliberatve}.Reasoning \\
& + 50\omega^2 : \text{ReasoningReactive}.Reasoning \\
& + 17\omega^2 : \text{ProcessingSortingAndStorage}.Processing \\
& + 17\omega^2 : \text{ProcessingGeneration}.Processing \\
& + 17\omega^2 : \text{ProcessingPrediction}.Processing \\
& + 16\omega^2 : \text{ProcessingDiagnosis}.Processing \\
& + 16\omega^2 : \text{ProcessingRecovery}.Processing \\
& + 17\omega^2 : \text{ProcessingRemediation}.Processing
\end{aligned}
$$

$$n\omega^{k+l} + m\omega^k = n\omega^{k+l} = m\omega^k + n\omega^{k+l}$$

$$n\omega^k + m\omega^k = (n+m)\omega^k = m\omega^k + n\omega^k$$

Agent State	Actions leading to the agent state	f	p
	Identity		
Communicating	SendMessageWorker	50	2
	SendMessageLeader	50	2
	SendMessageError	1	1
	ReceiveMessageWorker	50	2
	ReceiveMessageLeader	50	2
	ReceiveMessageError	1	1
Reasoning	ReasoningDeliberatve	50	2
	ReasoningReactive	50	2
Processing	ProcessingSortingAndStorage	17	2
	ProcessingGeneration	17	2
	ProcessingPrediction	17	2
	ProcessingDiagnosis	16	2
	ProcessingRecovery	16	2
	ProcessingRemediation	17	2

Specification

```
AEIP {
   MESSAGES { ... }
   CHANNELS { ... }
   FUNCTIONS { ... }
   MANAGED_ELEMENTS {
      MANAGED_ELEMENT worker {
         INTERFACE_FUNCTION getDistanceToNearestObject { RETURNS { DECIMAL } }
      }
   }
} // AEIP

METRICS {
   METRIC distanceToNearestObject  {
      METRIC_TYPE  {  RESOURCE  }
      METRIC_SOURCE { AEIP.MANAGED_ELEMENTS.worker.getDistanceToNearestObject }
      DESCRIPTION  { "measures the distance to the nearest space object" }
      MEASURE_UNIT  { "KM" }
      VALUE { 100 }
      THRESHOLD_CLASS  {  DECIMAL  [0.001 ~ )  }
   }
}
```

Autonomic Computing

Inspiration from the human/mammalian autonomic nervous system.

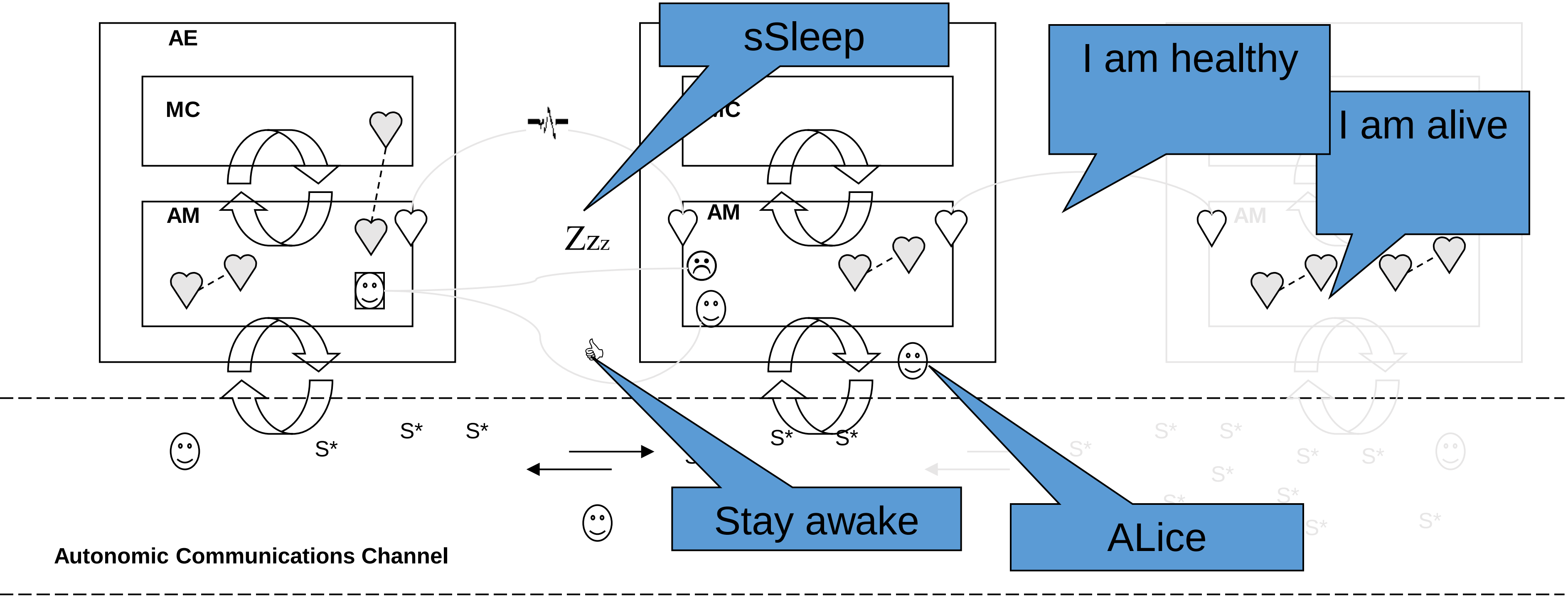
Fight or Flight

sympathetic
(SyNS)

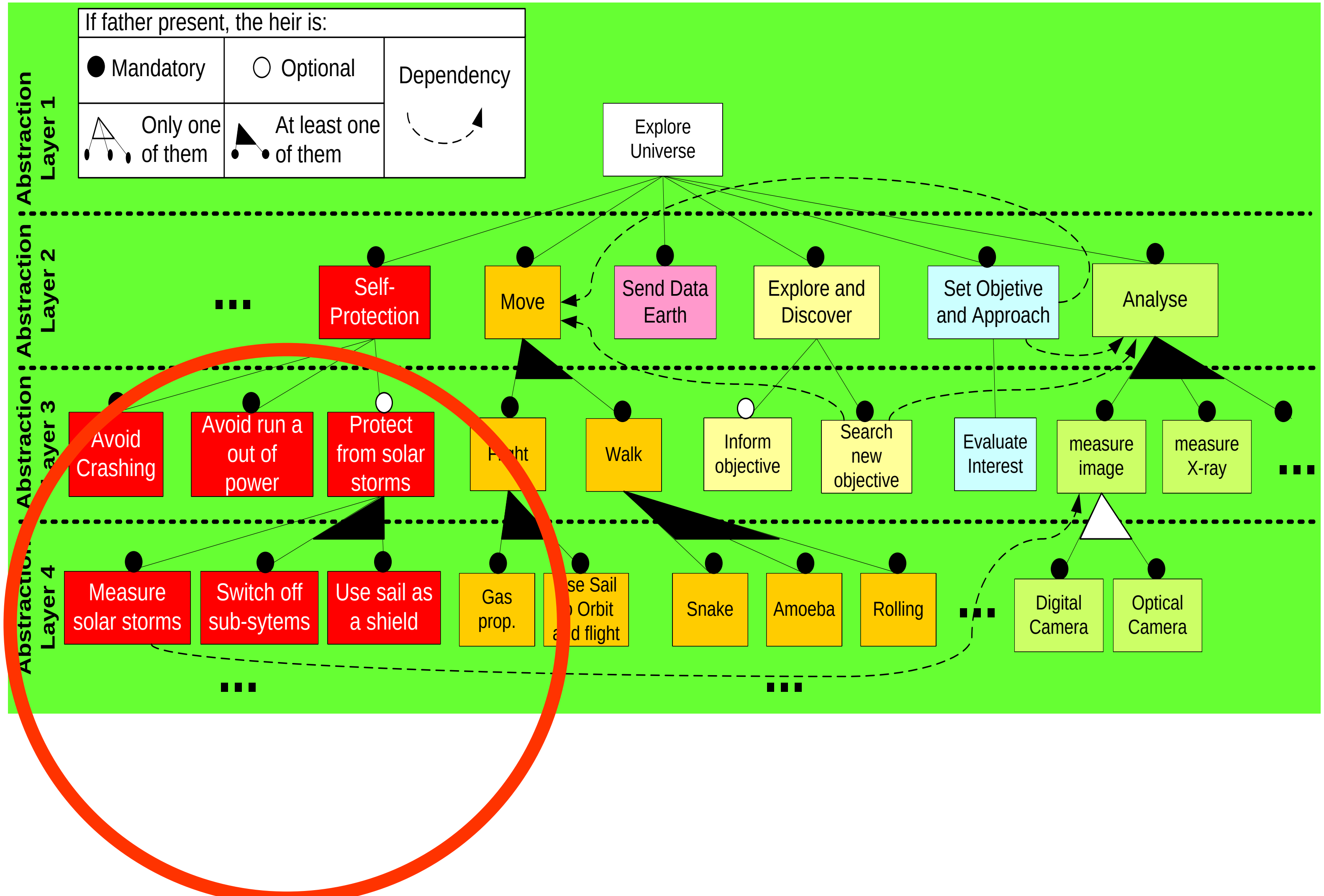
Rest and Digest

parasympathetic
(PaNS)

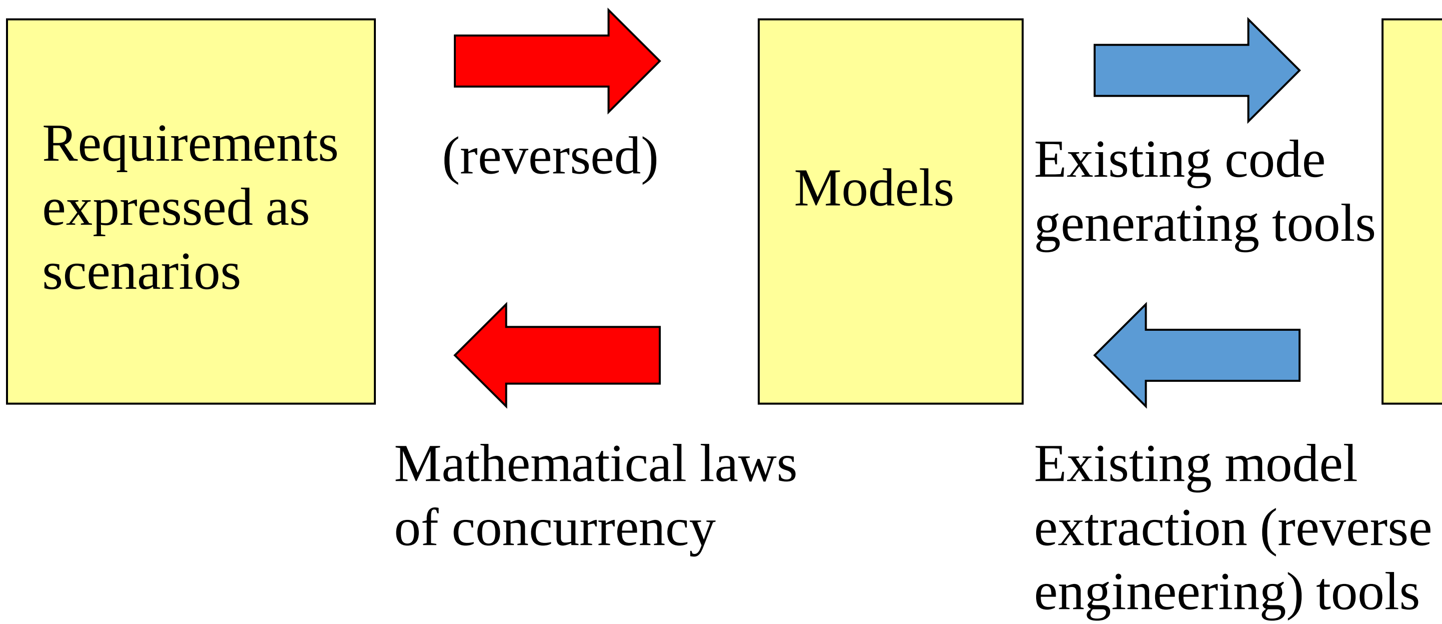
Autonomic Environment



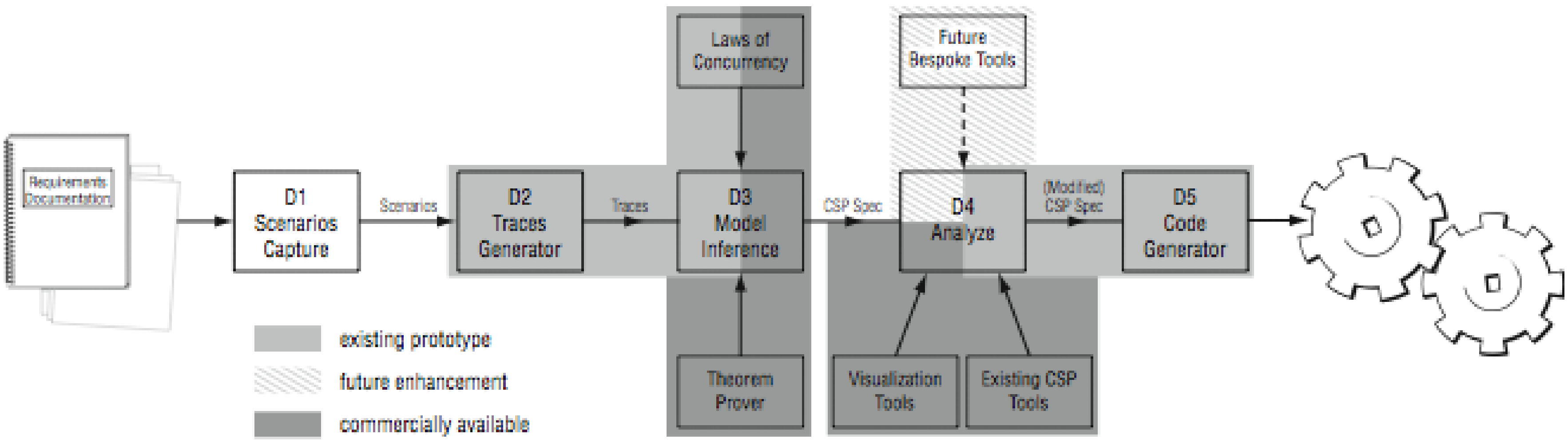
SPL / Feature Model



Requirements to Design to Code (R2D2C)



Current Status



Benefits of the Method

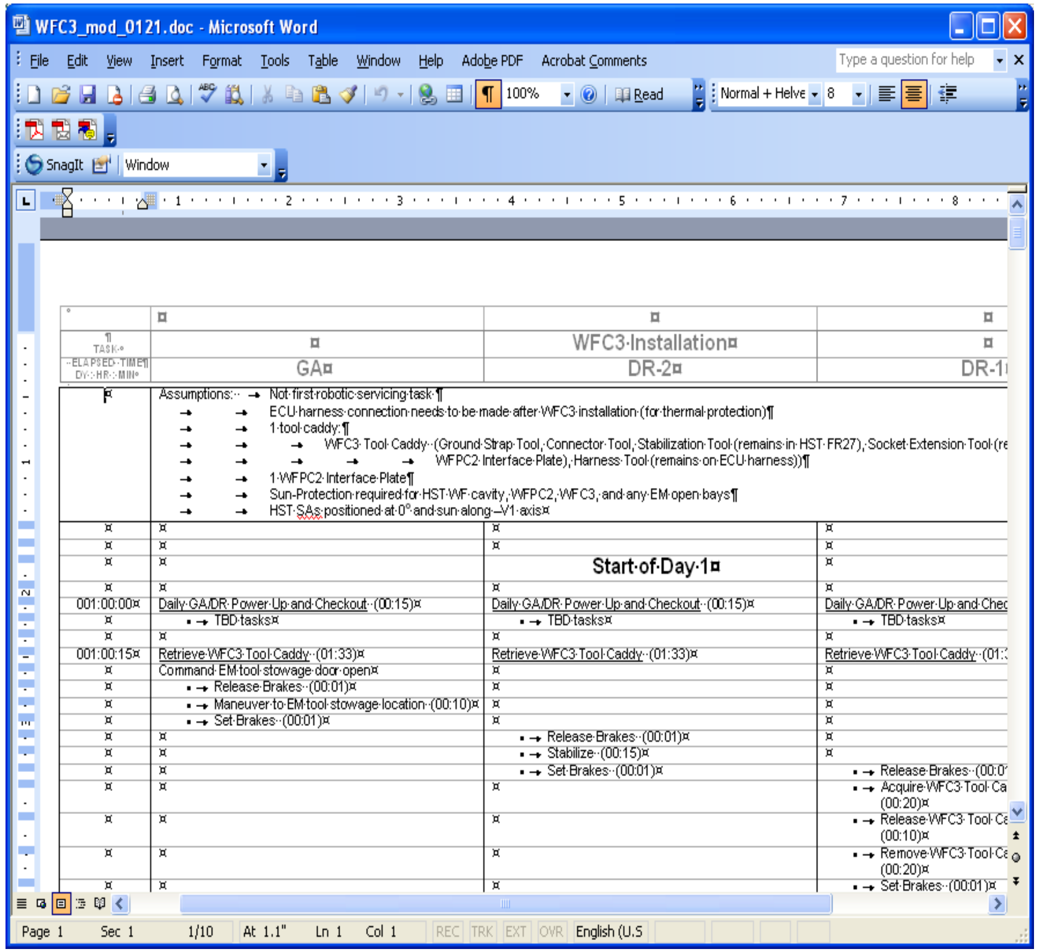
- Automation of entire development process;
- Significant increase in quality;
- Ability to do formal proof on properties of implementations;
- Ability to do formal proof of correctness;
- Automated means for requirements analysis;
- Guaranteed correspondence between requirements and their implementation as code.

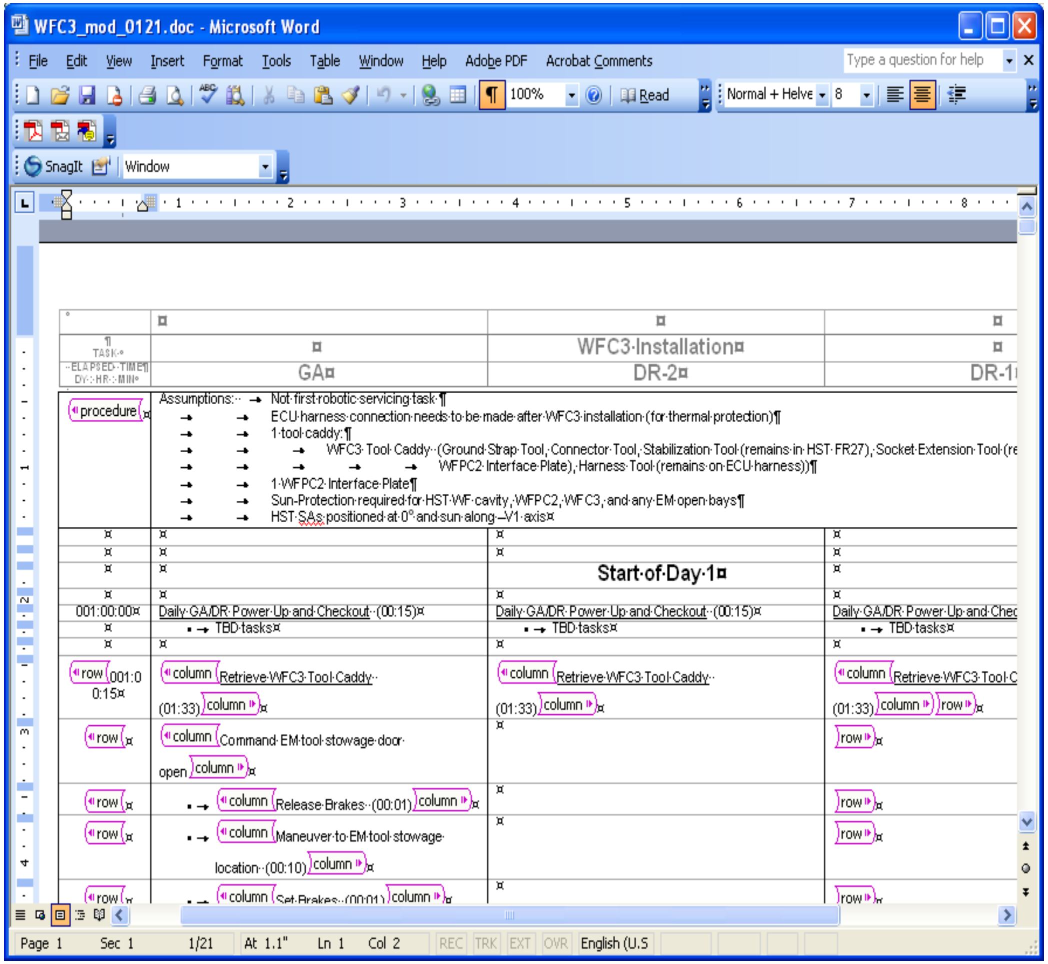
Applications

- End-to-end automatic code generation of provably correct systems;
- Automatic reimplementation after any requirements change;
- Exploiting re-use across platforms;
- Reverse engineering legacy systems to a mathematically sound model;
- Analysis and documentation of existing systems (e.g., expert systems);
- Re-engineering of legacy systems to a provably correct new implementation.

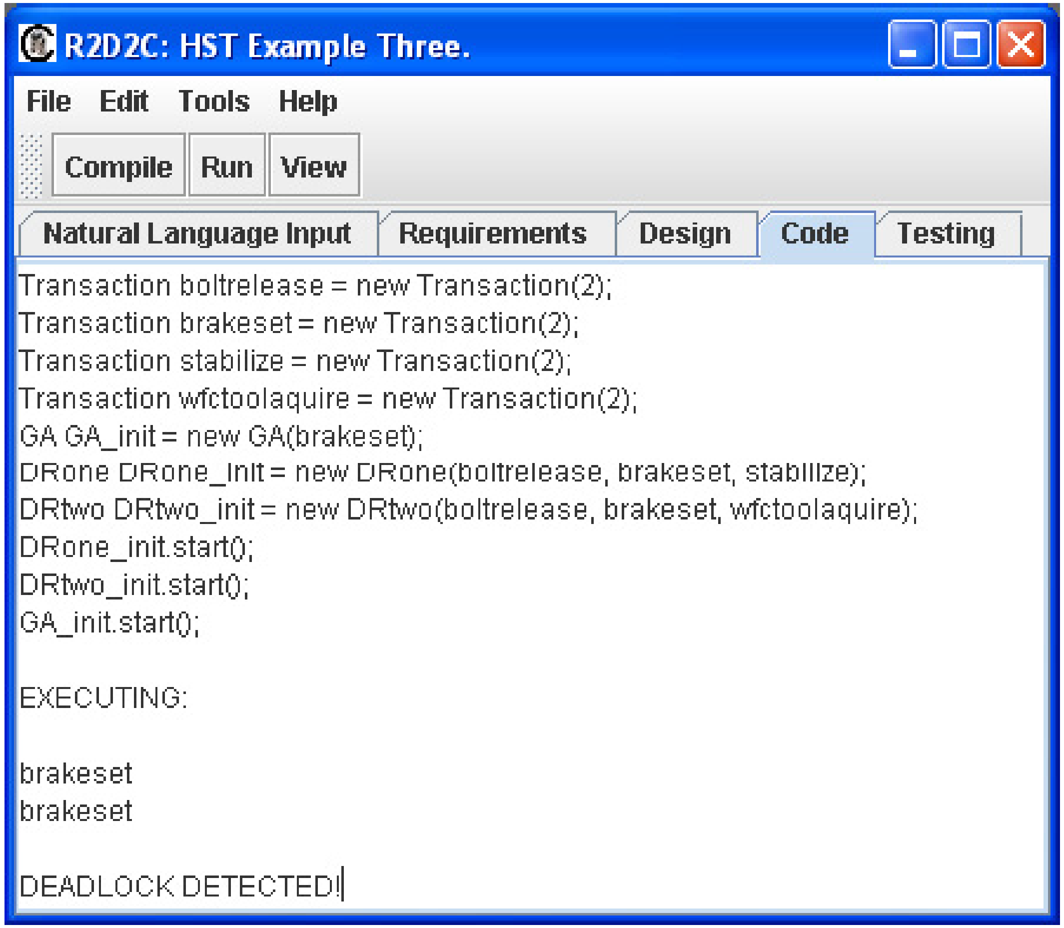
Domains (to date)

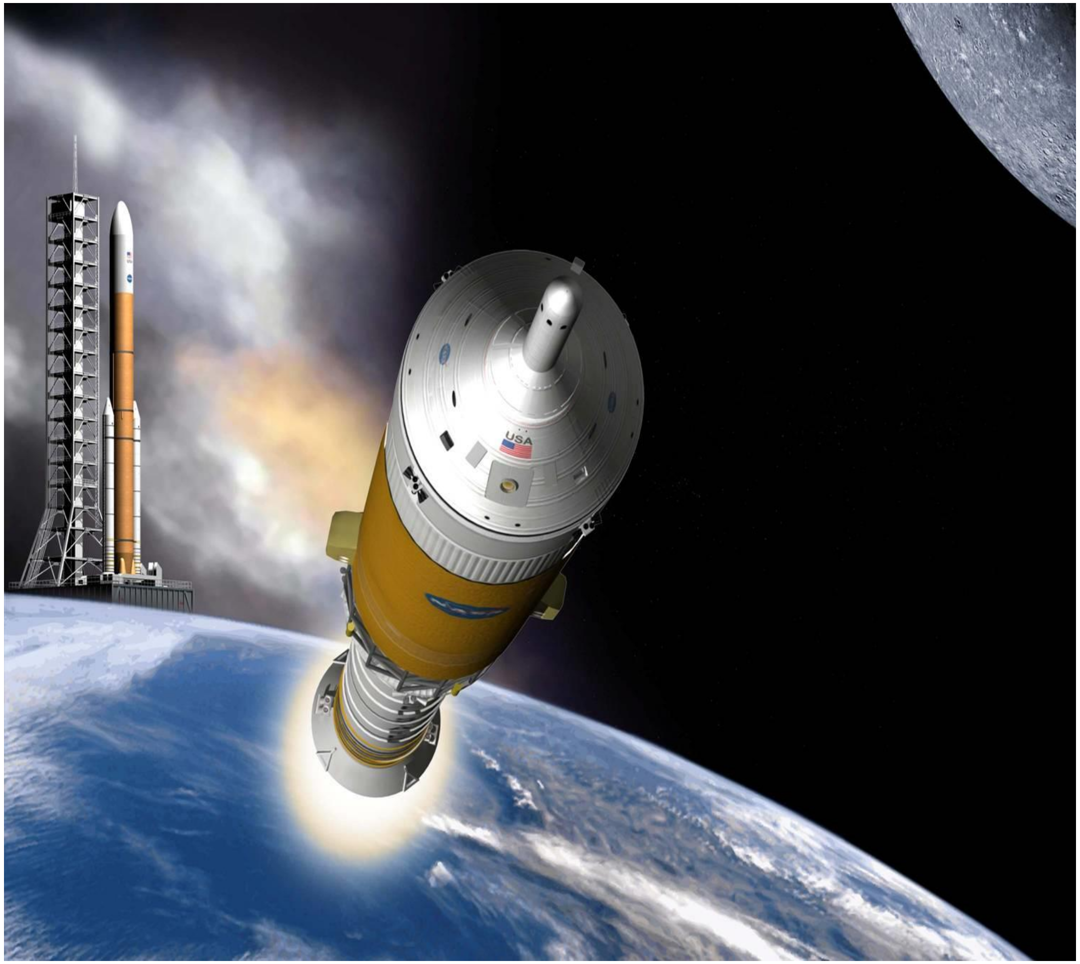
- Agent Based Systems;
- Wireless Sensor Networks ;
- ANTS;
- Verification of Robotic Procedures (cf. Hubble Space Telescope Robotic Servicing Mission).


WFC3_mod_0121.doc - Microsoft Word
File Edit View Insert Format Tools Table Window Help Adobe PDF Acrobat Comments
Type a question for help
100% Read Normal + Helve 8
SnagIt Window
TASK ELAPSED TIME DY:HR:MIN
WFC3 Installation
GA
DR-2
DR-1
Assumptions: Not first robotic servicing task
ECU harness connection needs to be made after WFC3 installation (for thermal protection)
1 tool caddy:
WFC3 Tool Caddy (Ground Strap Tool, Connector Tool, Stabilization Tool (remains in HST FR27), Socket Extension Tool (re
WFPC2 Interface Plate), Harness Tool (remains on ECU harness))
1 WFPC2 Interface Plate
Sun-Protection required for HST WF cavity, WFPC2, WFC3, and any EM open bays
HST SAs positioned at 0° and sun along –V1 axis
Start of Day 1
001:00:00
Daily GA/DR Power Up and Checkout (00:15)
TBD tasks
001:00:15
Retrieve WFC3 Tool Caddy (01:33)
Command EM tool stowage door open
Release Brakes (00:01)
Maneuver to EM tool stowage location (00:10)
Set Brakes (00:01)
Stabilize (00:15)
Acquire WFC3 Tool Ca
(00:20)
Release WFC3 Tool Ca
(00:10)
Remove WFC3 Tool Ca
(00:20)
Page 1 Sec 1 1/10 At 1.1" Ln 1 Col 1 REC TRK EXT OVR English (U.S

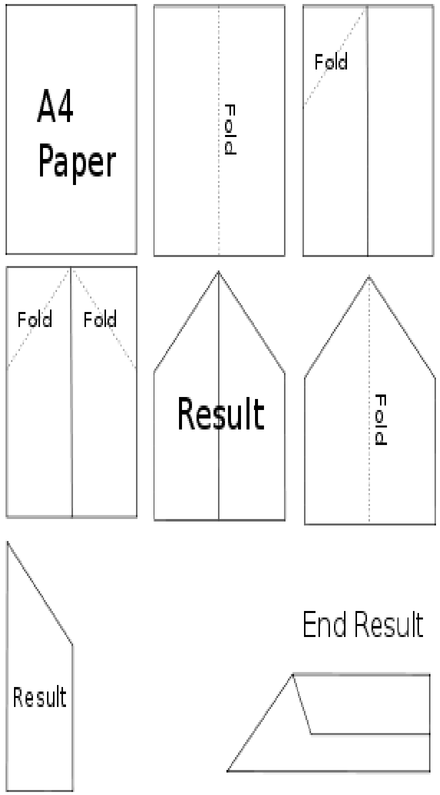

WFC3_mod_0121.doc - Microsoft Word
File Edit View Insert Format Tools Table Window Help Adobe PDF Acrobat Comments
Type a question for help
100% Read Normal + Helve 8
SnagIt Window
TASK
ELAPSED TIME
DY : HR : MIN
WFC3 Installation
GA
DR-2
DR-1
procedure
Assumptions: Not first robotic servicing task
ECU harness connection needs to be made after WFC3 installation (for thermal protection)
1 tool caddy:
WFC3 Tool Caddy (Ground Strap Tool, Connector Tool, Stabilization Tool (remains in HST FR27), Socket Extension Tool (re
WFPC2 Interface Plate), Harness Tool (remains on ECU harness))
1 WFPC2 Interface Plate
Sun-Protection required for HST WF cavity, WFPC2, WFC3, and any EM open bays
HST SAs positioned at 0° and sun along –V1 axis
Start of Day 1
001:00:00
Daily GA/DR Power Up and Checkout (00:15)
TBD tasks
row 001:00:15
column Retrieve WFC3 Tool Caddy (01:33) column
column Command EM tool stowage door open column
column Release Brakes (00:01) column
column Maneuver to EM tool stowage location (00:10) column
column Set Brakes (00:01) column
Page 1 Sec 1 1/21 At 1.1" Ln 1 Col 2 REC TRK EXT OVR English (U.S


R2D2C: HST Example One.
File Edit Tools Help
Compile Run View
Natural Language Input Requirements Design Code Testing
channel brakerelease, brakeset, stabilize, wfctoolaquire : T ;
GA = brakeset ! 0 -> GA ;
DRone = brakeset ? x -> brakerelease ! 0 -> stabilize ! 0 -> brakeset ! 0 -> DRone ;
DRtwo = brakeset ? x -> wfctoolaquire ! 0 -> brakeset ! 0 -> DRtwo ;
System = DRone [| {| |} |] DRtwo [| {| |} |] GA ;

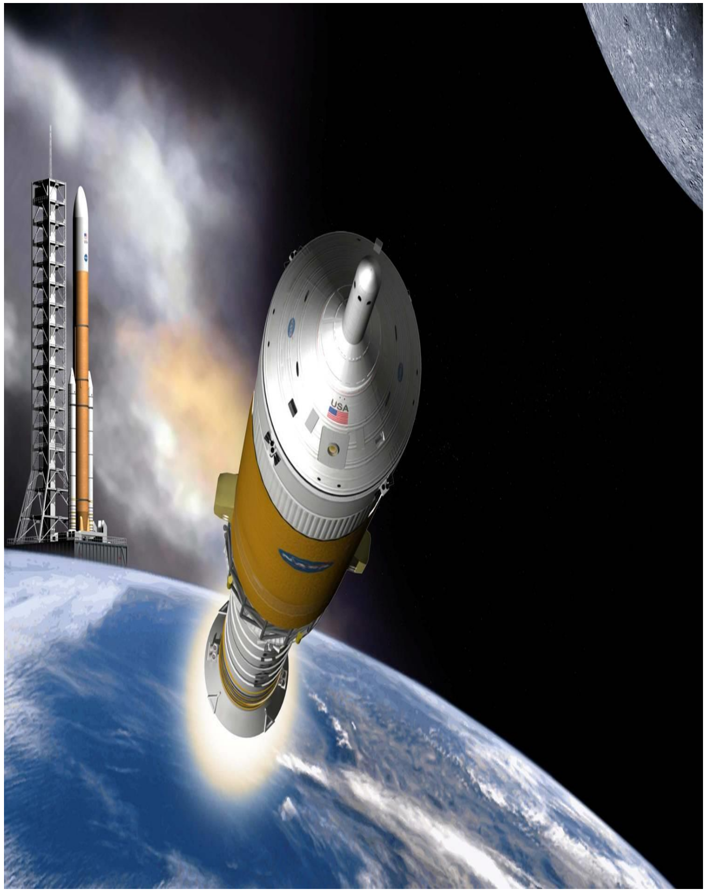

R2D2C: HST Example Three.
File Edit Tools Help
Compile Run View
Natural Language Input Requirements Design Code Testing
Transaction boltrelease = new Transaction(2);
Transaction brakeset = new Transaction(2);
Transaction stabilize = new Transaction(2);
Transaction wfctoolaquire = new Transaction(2);
GA GA_init = new GA(brakeset);
DRone DRone_init = new DRone(boltrelease, brakeset, stabilize);
DRtwo DRtwo_init = new DRtwo(boltrelease, brakeset, wfctoolaquire);
DRone_init.start();
DRtwo_init.start();
GA_init.start();
EXECUTING:
brakeset
brakeset
DEADLOCK DETECTED!


USA

Caveat



$\not\Rightarrow$



Conclusions

- Software must evolve.
- There is a tension between reliability, predictability and cost and this need for evolution.
- There is a need for an Evolving Critical Systems research effort.
- Lero and others are driving that effort.

Any problem in computer science can be solved with another layer of indirection.

But that usually will create another problem.

David Wheeler

lero THE IRISH SOFTWARE RESEARCH CENTRE

Go raibh maith agat!
Thank you!

Science Foundation Ireland
Ireland's European Structural and Investment Funds Programmes 2014-2020
Co-funded by the Irish Government and the European Union
European Union
European Regional Development Fund
SOUTHERN Regional Assembly
Promoting our Region

ACM: The Learning Continues…

- Questions about this webcast? learning@acm.org
- ACM Learning Webinars (on-demand archive): http://learning.acm.org/webinar
- ACM Learning Center: http://learning.acm.org
- ACM Queue: http://queue.acm.org/