WWW.IARIA.ORG

PANEL SOFTENG

Software Challenges for New Services/Devices

Petre DINI

Concordia University, Canada

China Space Agency Center, China

IARIA Organization

petre@iaria.org

New Paradigms

- Smart cities
- Smart devices
- Smart technologies
- eGovernment online services
    - → Urban computing, apps, embedded software
- Approximate computing [Big Data]
- Eventual consistency [Replicated Data]
    - → Storage/computation, data centers management
- Programming paradigms [agent-, context-, network-coding]
- Designing/development paradigms [agile, crowdsourcing]
- Machine learning [knowledge, semantic, taxonomies,]
- etc.

Changes in Technology/Service Lifecycle

Agile methods

Open source

Crowdsourcing

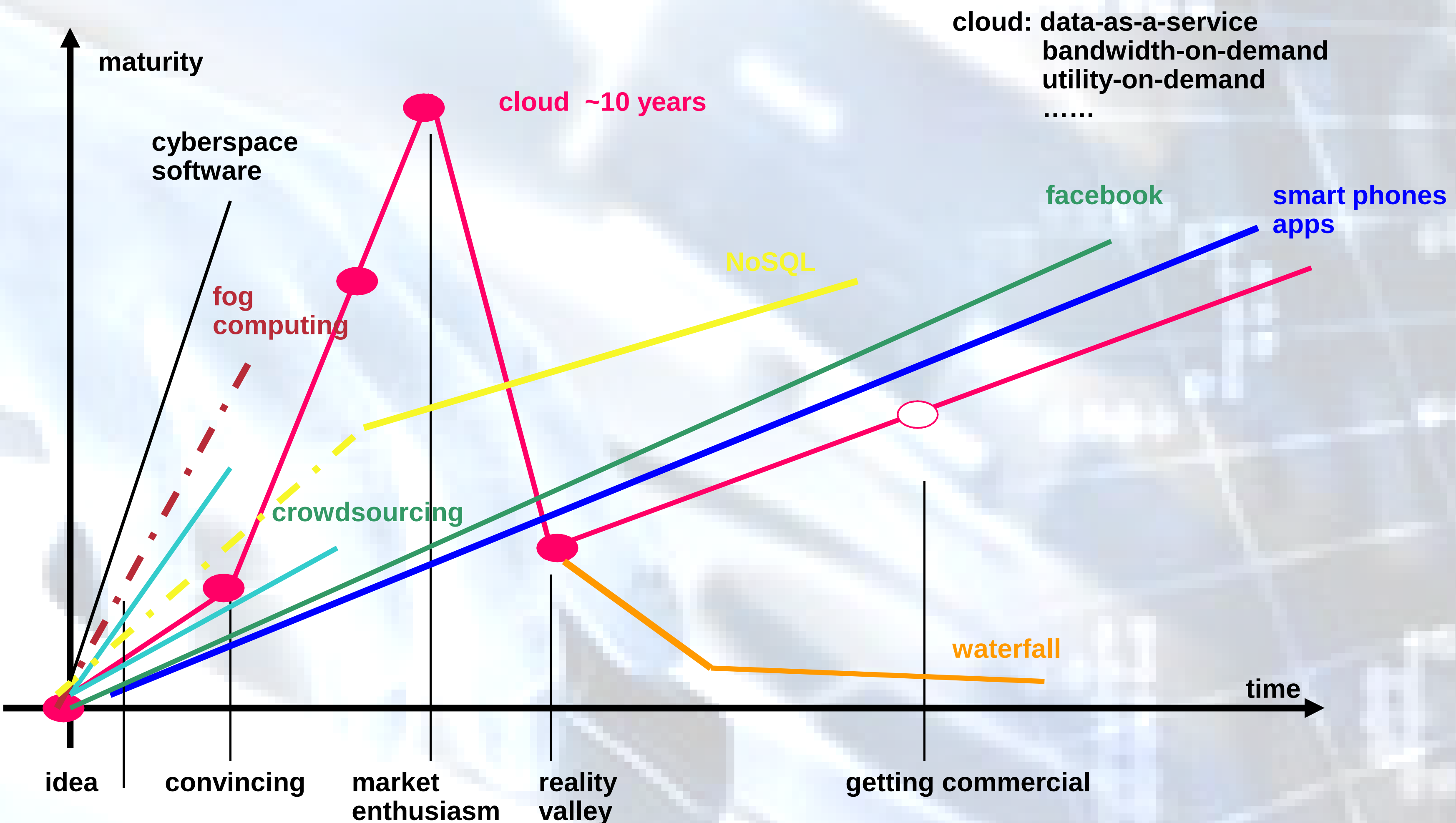
Design security/privacy-driven

Design for accessibility

Adaptive interfaces

….

Technology/Maturity Lifecycle



Today’s Panelists

- **Moderator:**
  Petre Dini, Concordia University, Canada | China Space Agency Center, China
- **Panelists**
  Laura Semini, Università di Pisa, Italy
  > software product lines
- **Michael Jäger, Technische Hochschule Mittelhessen, Germany**

  > security- and privacy-aware systems …software manipulations
- **Stepan Orlov, St. Petersburg State Polytechnic University, Russian Federation**

  > scaling the server, using clouds, and client-side computing

Q & A

Qs & As

WWW.IARIA.ORG

S. Orlov, N. Shabrov

SOFTENG 2015, Barcelona, Spain

Opportunities for non-commercial web applications to use computational resources

St. Petersburg Polytecnical University

Computer Technology in Engineering Dept.

http://equares.ctmech.ru

Background information

- More and more software is designed in the form of *Web applications* accessed through a regular Web browser
- *Simulation software* is not an exception
  - https://simscale.com
  - http://www.wolfram.com/mathematica
  - http://www.wolframalpha.com/
  - https://insightmaker.com/
- We've just added our own
  - **http://equares.ctmech.ru/**

commercial, runs simulations on server (simscale.com, wolfram.com/mathematica, wolframalpha.com)

non-commercial, simulations on client (insightmaker.com)

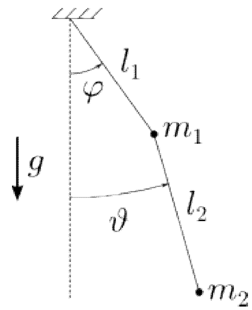
non-commercial, simulations on server (equares.ctmech.ru)

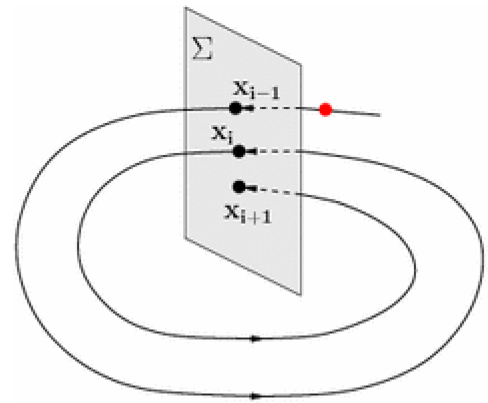
Background information

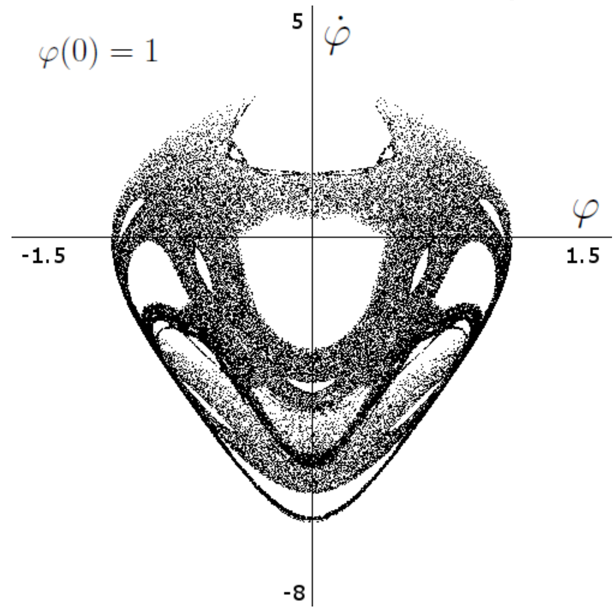
- Why yet another simulation software?
  - Something important is missing in other tools
- What we need
  - *Ability to formulate algorithm for numerical experiment in a flexible way*
  - *Efficiency*
    - Considerably faster, than MATLAB/SciLab
    - Ability to avoid extra memory usage in algorithms
  - *Accessibility*
    - Web-application running in browser
  - *Ease of use*
    - No algorithm programming. Instead, draw a scheme of data flows between data processing units
    - Ability to use an existing scheme and slightly modify it if necessary
  - *Extensibility*
    - Open source code
    - Everyone can add missing features that he/she needs

Background information

- Example 1 – double pendulum
  - There are quasi-periodic and chaotic motions
  - To see that, one should build the Poincare map
  - But it's too difficult in
    - MATLAB, SciLab
    - Mathematica (before version 9)
    - Maple
  - In any system (except XPP) programming is necessary, and sometimes the solver has to be re-implemented
  - And it will work too slow





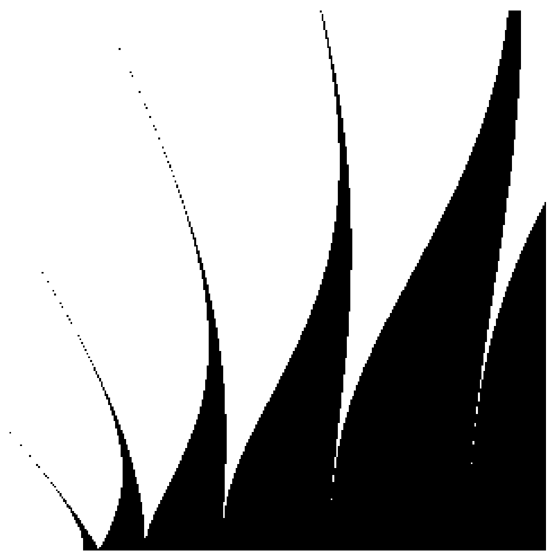


Background information

- Example 2 – Mathieu equation

$$\ddot{q} + [\lambda - 2\gamma\cos(2q)]\,q = 0$$

- Ince – Strutt diagram shows parametric resonance instability areas on plane of parameters
- How student can obtain it?
  - Analytically – too difficult
  - Numerically – well, the fundamental matrix has to be computed at each pixel, and its eigenvalues – a bit too difficult
  - And it will work too slow if we use MATLAB/SciLab

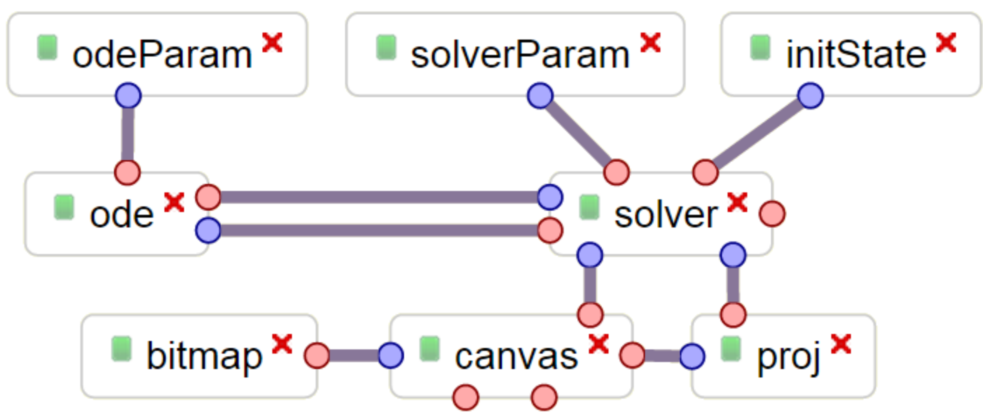


Choice of technologies

- Server
  - Node.js (web-server)
  - C++, Qt (computational core running on server)
  - MongoDB (server database)
  - Jade (HTML generation engine)
- Client
  - HTML 5 (using SSE)
  - D3 (interactive visualization of schemes)
  - MathJax (formulas in documentation)
  - CSS, JavaScript, jQuery, jQueryUI (common things)

Simulation specification

- There is a scheme consisting of *boxes* and *links* (pipes-and-filters design pattern)
  - Links determine data flows
  - Boxes are data processors
  - A box has *ports*; there are *input* and *output* ports
  - A link connects an output port of a box and an input port of some other box
  - Boxes without input ports are *data sources*
  - Boxes without output ports are *data storage*
  - A box can have *parameters*



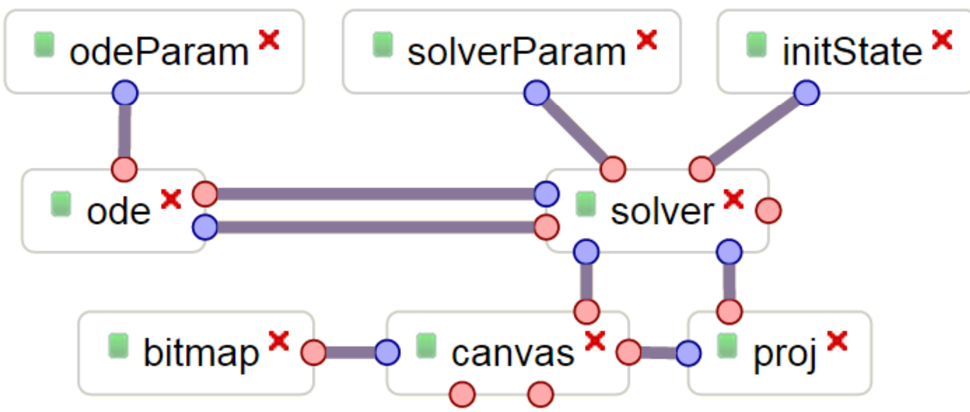
Simulation specification

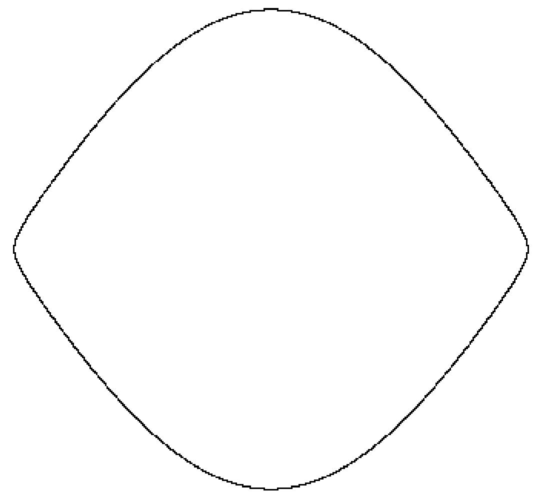
- Simulation scheme defines a *data processing system* implementing numerical experiment
- User draws simulation scheme in browser and sends it to server as JSON

{"boxes":[{"name":"odeParam","type":"Param","props":{"data":["1","1","0.6","1","9.8"]}},{"name":"solverParam","type":"Param","props":

Examples

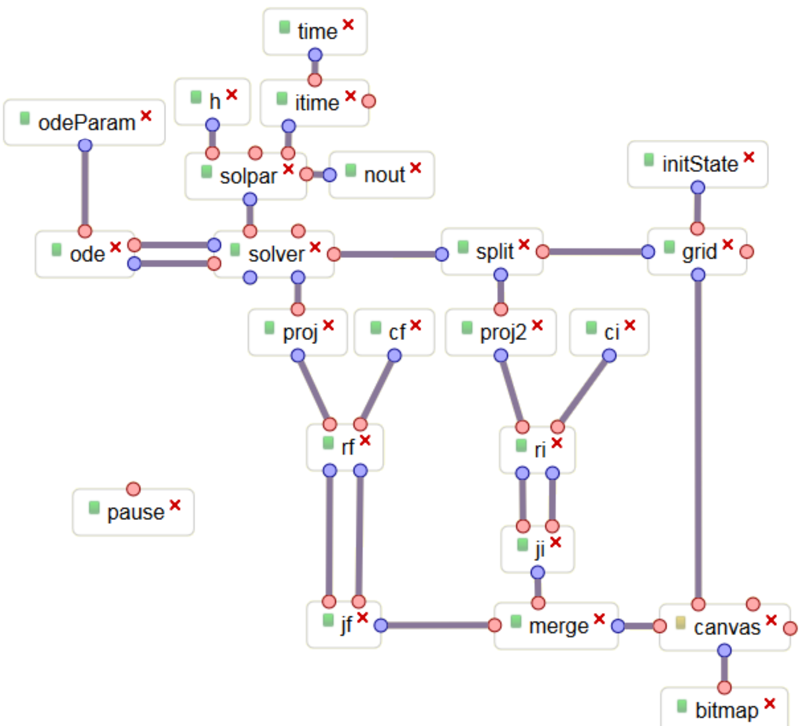
- Simple pendulum, one phase curve

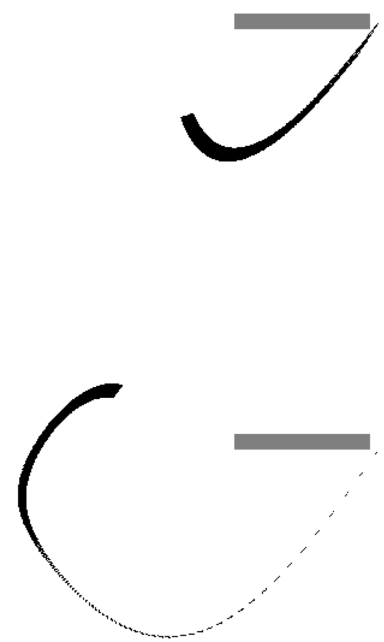




Examples

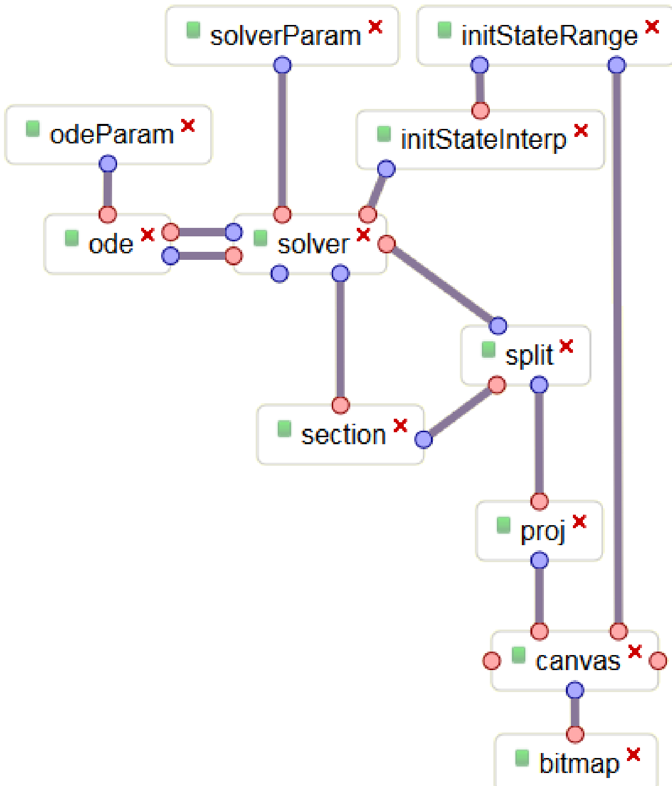
- Simple pendulum, phase volume evolution

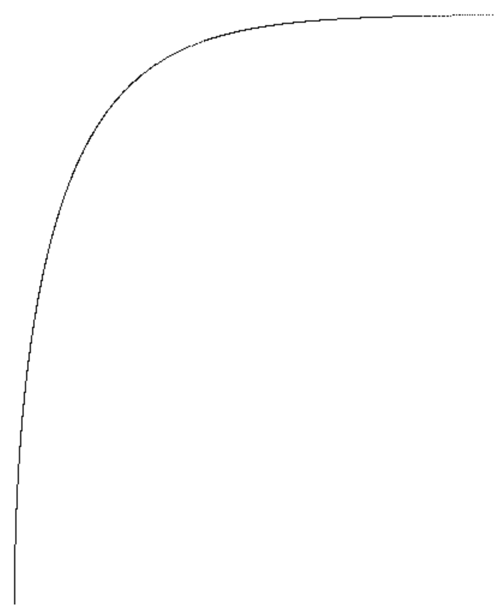




Examples

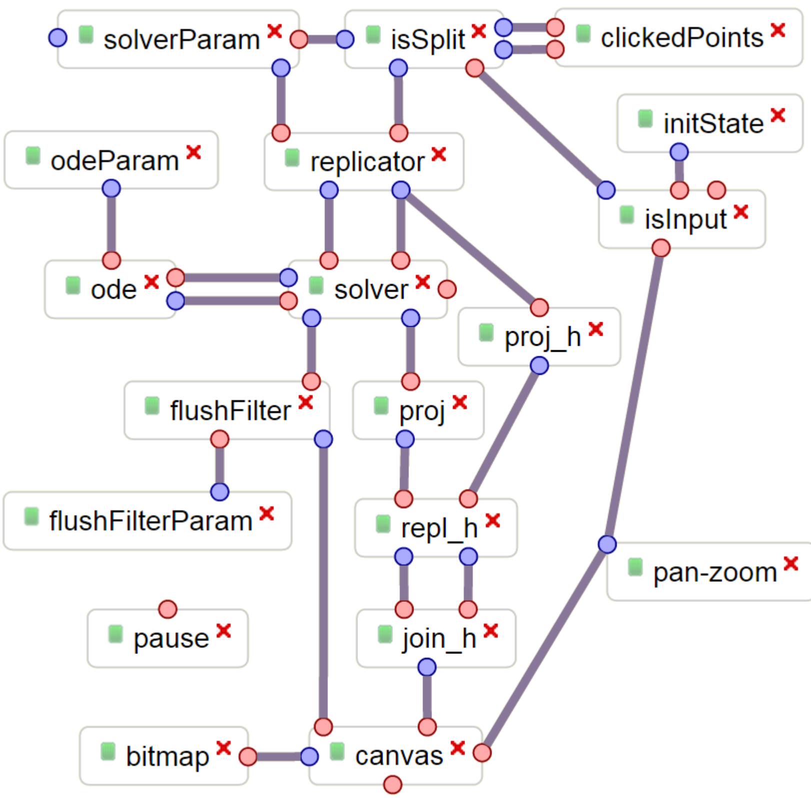
- Simple pendulum, skeletal curve

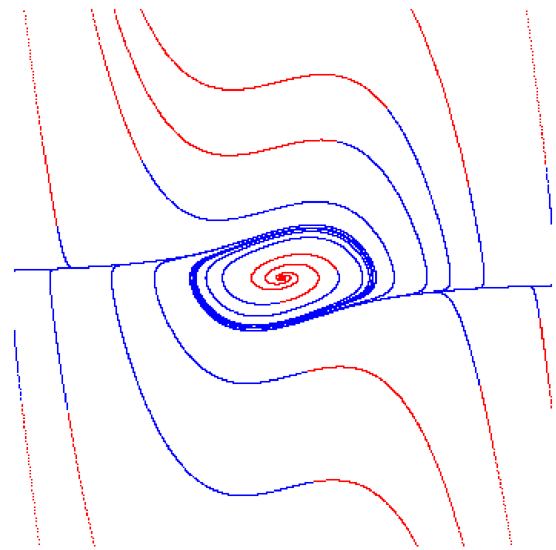




Examples

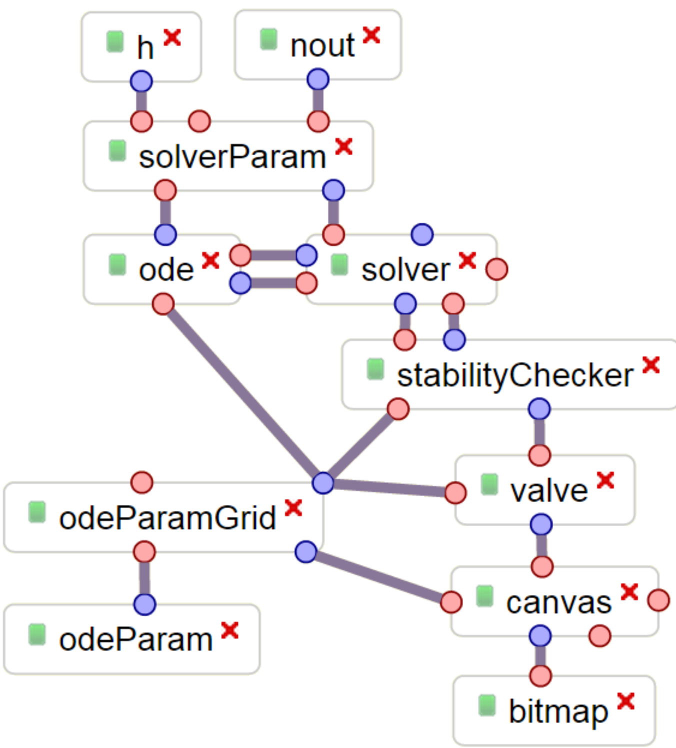
- Interactive phase portrait

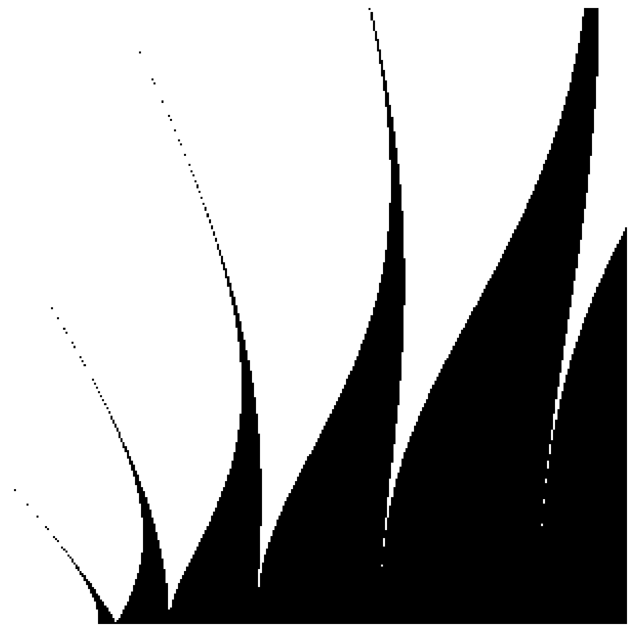




Examples

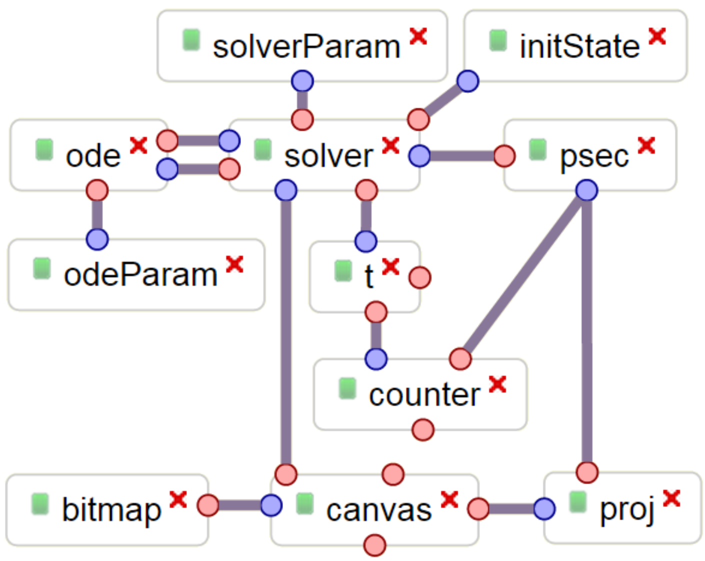
- Ince – Strutt diagram (500 x 500, 6.3 s)





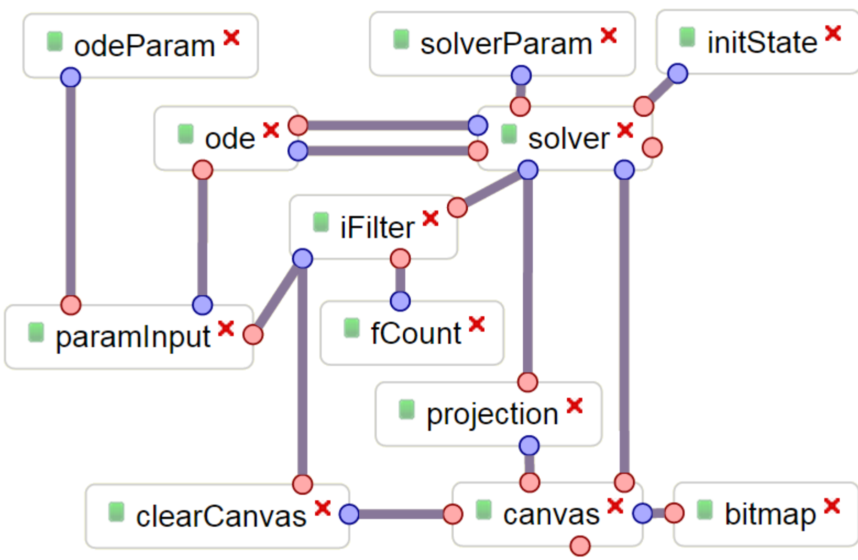
Examples

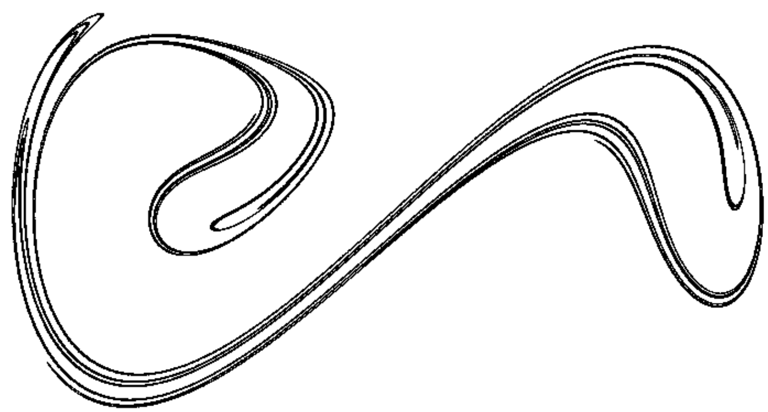
- Double pendulum, Poincare map
  - 50000 points, 28.5 s



Examples

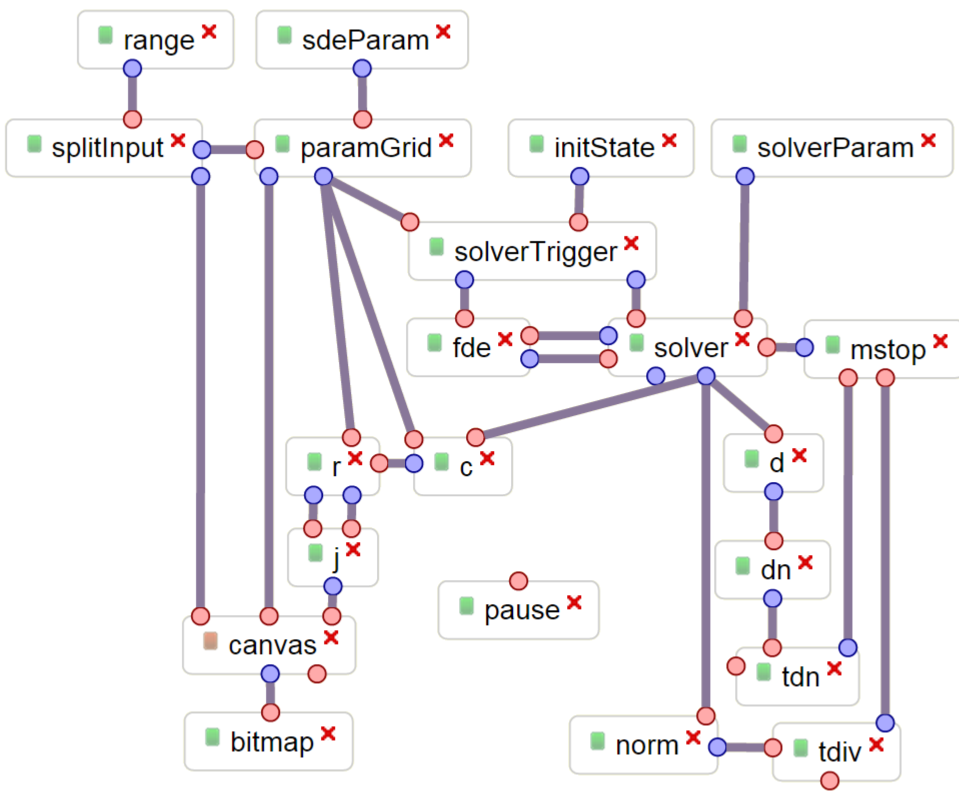
- Strange attractor
  - Forced Duffing oscillator
  - Interactive parameter change

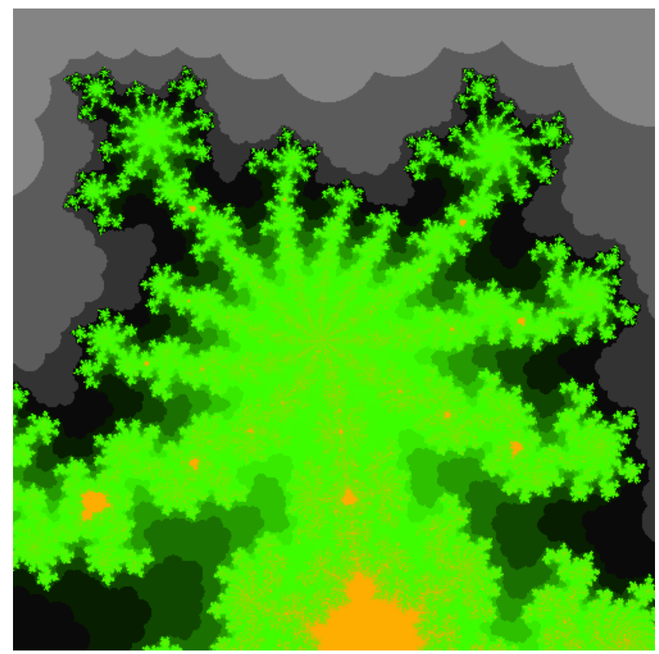




Examples

- Colored Mandelbrot set
  - Interactive pan/zoom

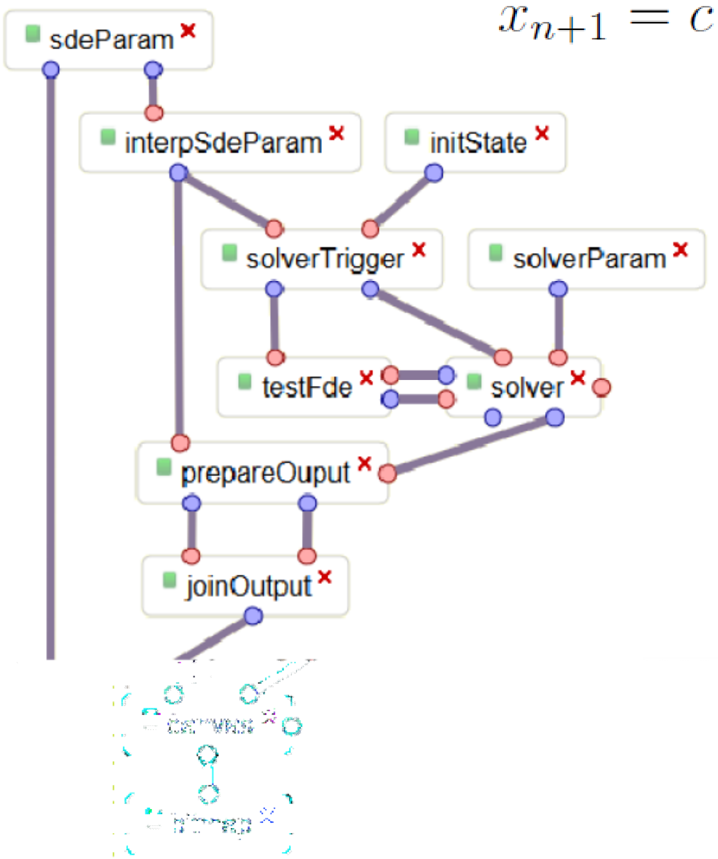


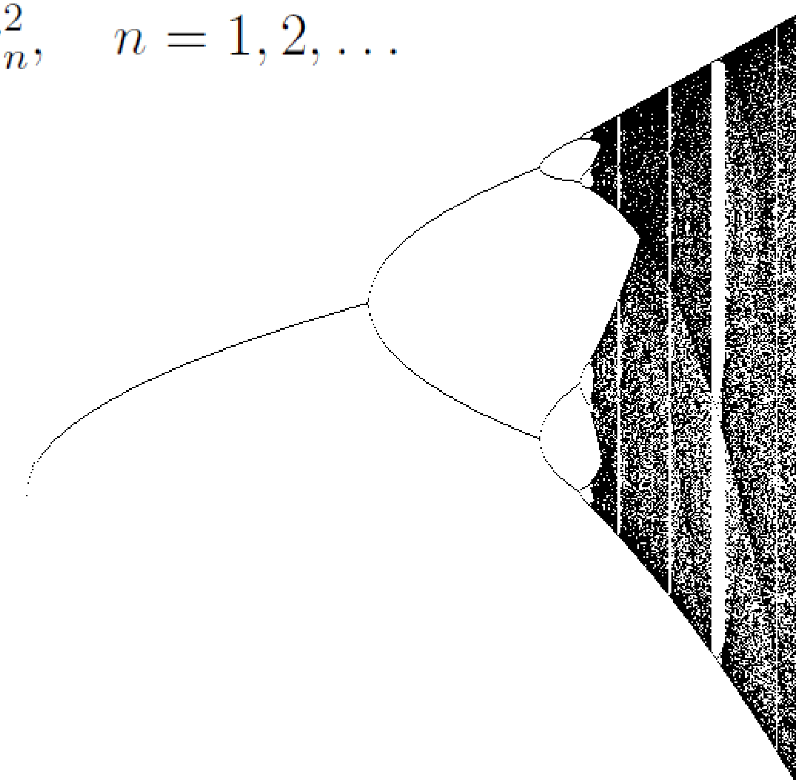


Examples

- Chaos in logistic mapping

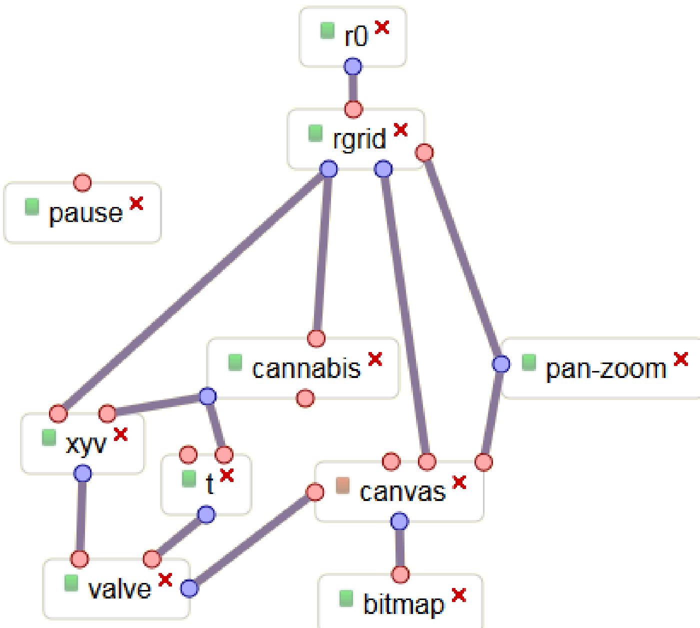
$$x_{n+1} = c - x_n^2, \quad n = 1, 2, \ldots$$





Examples

- Set of points where a function has a positive value
  - http://habrahabr.ru/post/135344/



Scalability problem

- Simulations run on server
- Each client can consume a thread for as long as user wants
  - Some simulations finish fast
  - Some run for a long time
- Currently web server is one 16-cpu node
- What if our it will become popular?

Scalability problem

- Possible solutions
  - Add more servers
    - Who will buy them?
  - Limit user resources on server
    - Limit memory – ok
    - Limit CPU time – won't help much
      (user may really need ~1 minute, or maybe more)
  - Move simulations to client
    - Back to desktop applications?
    - Use JavaScript or asm.js? But we're using LAPACK and will add other 3^{rd} party libs
  - Cloud computing
    - Who will pay? Or can it be available for free?

Thank you!

Questions?

http://equares.ctmech.ru

Software Product Lines
&
Feature Oriented Systems

Behaviour analysis

Laura Semini

Dipartimento di Informatica

Università di Pisa, Italia

Software product lines

- A Software Product Line is a family of software products that:
  - share a common architecture
  - varies in the functionalities offered
- Product line engineering accelerates product development by
  - Exploiting the commonalities among the product line members while
  - Managing the differences (called variabilities)
- The definition of models to specify and analyse Software Product Lines is a current challenge
  - potentially exponential number of different products that can be obtained

Feature-oriented development

- Much of today software is developed using scenarios or use cases
  - Essentially, a use case is the specification of a feature
- Feature = Incremental unit of functionality [def by Pamela Zave]
- Moreover, features are a convenient way to model
  - commonalities and variabilities of Software Product Lines

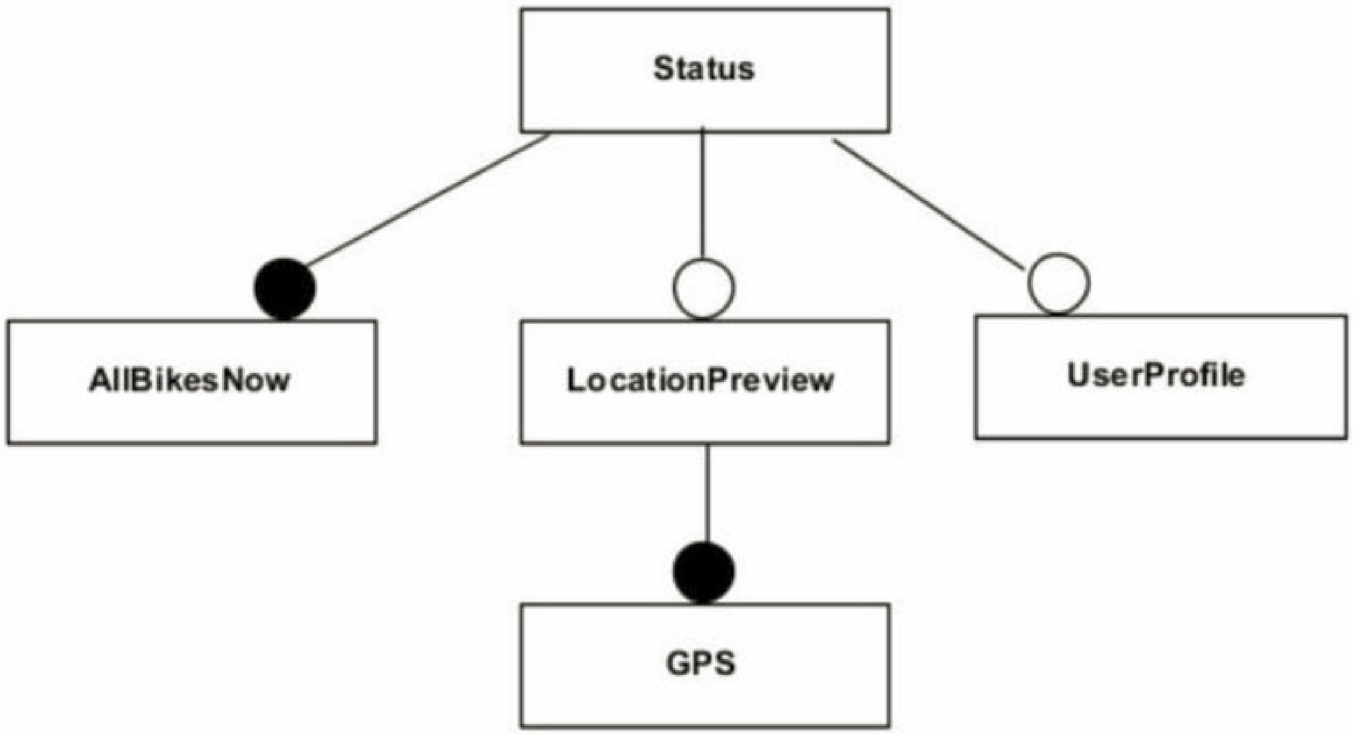
Feature-oriented systems: interactions

- Features may interact in unexpected ways
  - Initially addressed in telecommunication systems

- Features
  1. are often engineered by people who are not aware of the feature-interaction problem, and
  2. are used in safety-critical domains.

- Therefore:
  - (intended) feature interactions must be very well explained, and
  - we should not strive for solutions (for unintended interactions) that work only half the time.

Feature diagrams



Dimensions: design vs development perspectives

- Two essentially different perspectives on feature interaction:

1. Design:

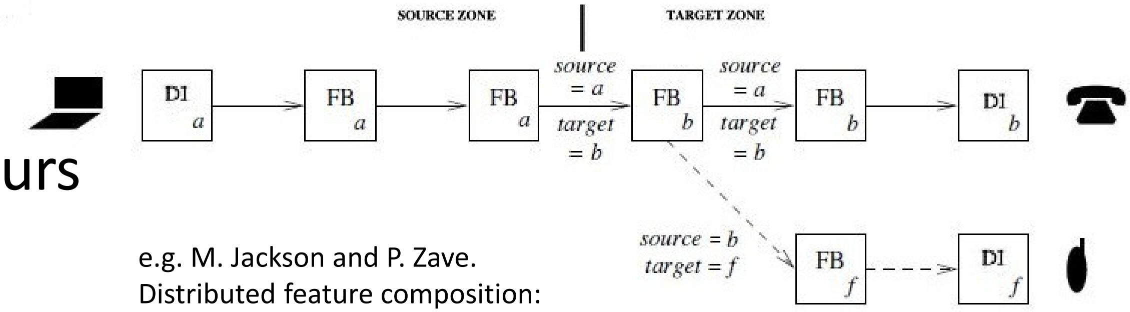
   Here the focus is on understanding requirements with a view towards building a system in which features either do not interact, or do so in a positive way.

2. Development:

   The focus here is on code, with a view towards analysing and understanding an existing system.

Features: annotation vs composition- based

- Composition-based
  - Composition of behaviours



e.g. M. Jackson and P. Zave.
Distributed feature composition:
a virtual architecture for telecommunications services. IEEE TSE 1998

- Annotation-based
  - Implemented with directives to the preprocessor

```
class Stack {

  boolean push(Object o) {
#ifdef LOCKING
    Lock lock = lock();
    if(lock == null) {
#ifdef LOGGING
      log("lock failed for: "+o);
#endif
      return false;
    }
#endif
#ifdef UNDO
    rememberValue();
#endif
    elementData[size++] = o;
    /*...*/
  }

#ifdef LOGGING
  void log(String msg) { /*...*/ }
#endif
#ifdef UNDO
  boolean undo() {
#ifdef LOCKING
    Lock lock = lock();
    if(lock == null) {
#ifdef LOGGING
      log("undo-lock failed");
#endif
      return false;
    }
#endif
    restoreValue();
    /*...*/
#ifdef LOGGING
    log("undone.");
#endif
  }

  void rememberValue() { /*...*/ }
  void restoreValue() { /*...*/ }
#endif
}
```

Fig. 9.2 Implementing a stack data structure with preprocessor directives; coordination code is implemented by nesting preprocessor directives

Open challenges

- Model behaviour in feature-based systems.
- Look for patterns of feature interaction.
- Designing a “feature-aware computational model”

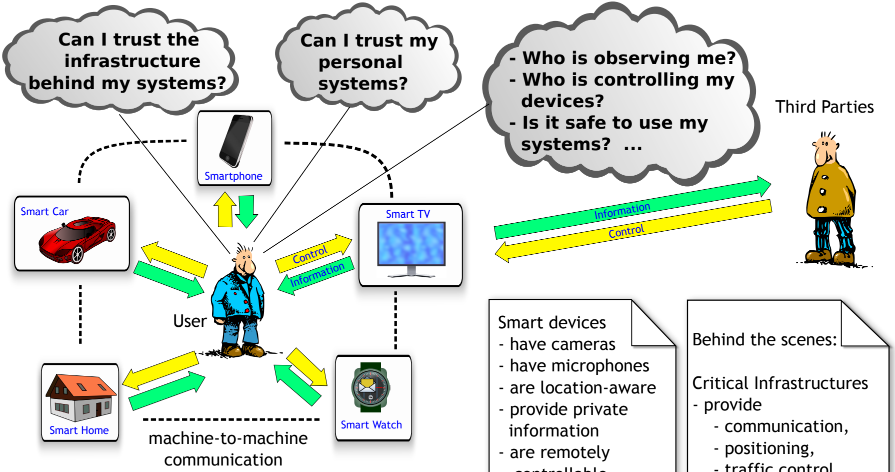
Trusted Ubiquitious Computing

Michael Jäger

Technische Hochschule Mittelhessen
michael.jaeger@mni.thm.de

SOFTENG, April 2015

A User-centric View on Ubiquitious Computing



Smart devices
- have cameras
- have microphones
- are location-aware
- provide private information
- are remotely controllable
- are exposed to
  - misuse
  - internet attacks

Behind the scenes:

Critical Infrastructures
- provide
  - communication,
  - positioning,
  - traffic control,
  - etc.
- are exposed to internet attacks.

Systems Engineering Challenge:
How can we build trustworthy systems?

What is Trust?

An entity can be trusted if it always behaves in the expected manner for the intended purpose (Trusted Computing Group).

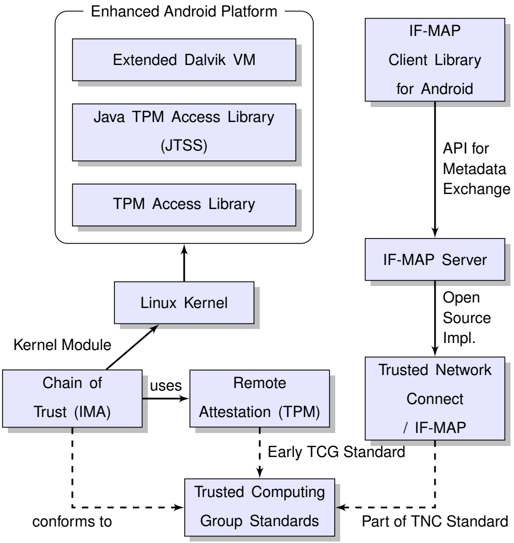
Claims

- Trust is a key factor for acceptance.
- Misuse and manipulation will happen increasingly.

How to build trusted systems?

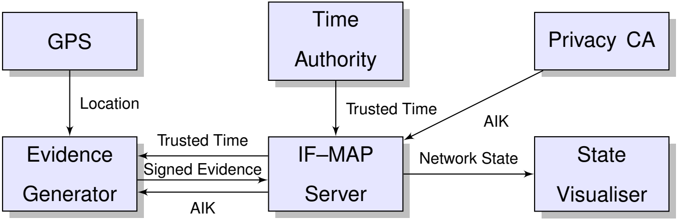
- **Security by Design**
  - Take cyber-attack and information misuse attempts for granted.
  - Model misuse cases, derive security requirements.
  - Be concerned about privacy.
- **Use trusted Components**
  - Secure network protocols
  - Open source software
  - Manipulation-protected key components: "Trusted Device"

Trusted Information Agent (TIA): Technical building blocks



(Coppolino, Jäger, Kuntze, Rieke: A Trusted Information Agent for Security Information and Event Management, ICONS 2012)

TIA architecture



(Coppolino, Jäger, Kuntze, Rieke: A Trusted Information Agent for Security Information and Event Management, ICONS 2012)