HolaCloud Roadmap

Lutz Schubert, University of Ulm

Keith Jeffery, Consultant

+ contributions from the conference
& cluster meetings

the modern environment

CLOUDS, SOFT-WARE, IOT ETC.

- **“Clouds” have become a synonymous term for “the internet”**, mostly because hardly any service or application provisioning would be economically possible without the elasticity of clouds
- **Embedded devices are everywhere**, ranging from smartphones over TVs and media players to washing machines, coffee makers etc. More and more of these devices are connected to the internet
- **Software pervades all** as more and more decision and control is moved from hardware to software, and as more and more of our modern devices (services, tasks, jobs) are software controlled.
- **“Data is the new oil”**…

the modern environment

BUT OUR SOFTWARE IS NOT READY...

- **Data is distributed and volatile** and no longer stored or available from a single location. Multiple users access the same data, manipulate it, share it etc.
- **Users are mobile** and not tied to one single location. What‘s available today may be unavailable tomorrow.
- **Devices and contexts constantly change and become more heterogeneous.** There is not only one processor or one device on the market, let alone one operating system, one middleware, one application...
- **The maintenance lifecycle becomes shorter and shorter** as devices and processors come and go, competitive services pop up and disappear etc. Modern providers struggle to keep up with the market.

the needs

MISSING CAPABILITIES

- **advanced systems development method(s)** that allow developers to express their actual intention of the code without having to develop the full algorithm by basing on pre-existing modules and patterns;
- **placement and locality of code and data** freely in a “mesh” of resources so as to ensure that they are at the right place at the right time so that the main incentives or quality criteria are always met;
- **adaptive management of software (self-adaptive code) and information** within context and requirements to ensure that intention and incentive are always met even when the context changes;
- **interoperability and portability** of the application over the “mesh” of resources by abstracting from code and data and instead acting on information and intention;

HOLA CLOUD

the needs

ADVANCED SYSTEMS DEVELOPMENT METHODS

- Based on model-driven technology, and making use of AI at development, respectively integrating it into execution level
- Model validation at development time to identify whether essential (functional and non-functional) requirements are met, e.g. security
- Native integration of network and edge/fog devices and capabilities with full mobility support

the needs

PLACEMENT AND LOCALITY

- Integrate different communication protocols and mobile interfaces right into the stack (remove responsibility from developer)
- Exploit network-based approaches where hierarchical clouds can come and go on demand / need
- Improve resilience and fault tolerance under the volatile conditions
- Enable adaptive partitioning of data and (de)composition of software on the fly
- Support provenance and curation
- Free placement of data, software, resources and users to ensure that data and services are always available to the end user(s)
- Deal with the 5V (volume, velocity etc.)

the needs

(SELF-)ADAPTIVE CODE (AND ENVIRONMENTS)

- Provide improved monitoring mechanisms that learn the relationship between algorithms / execution and requirements / quality of service
- Ensure that configuration and control pervades the whole software stack without generating conflicts within / between levels
- Enable reasoning with and about uncertainty and incompleteness
- Realise self-adaptive environments / code that meets the (business) requirements at run-time and can deal with the emergent behaviour in clouds

the needs

INTEROPERABILITY AND PORTABILITY

- Expand virtualisation technologies with the means to expose and hide platform characteristics on demand
- Define a formal syntax and (multilingual) semantics for integrating different standards and dialects
- Provide the means for interoperability and portability across hybrid CLOUD platforms and across heterogeneity of data and software, devices and users
- Go beyond standard means of virtualisation

consequences / approach

TOWARDS I3

an increased abstraction *without performance degradation* is necessary to improve interoperability, usability, portability …

requires strong rethinking of traditional methods

- **Most computation is about information, no longer about data**
  - Data doesn't always have to be exact
  - Data doesn't always have to be available
  - Computation doesn't always have to be complete

  ➔ *information aspect of the system*

consequences / approach

TOWARDS I3

an increased abstraction *without performance degradation* is necessary to improve interoperability, usability, portability …

requires strong rethinking of traditional methods

- **The “right” choice of deployment, configuration, computation etc. depends on too many factors**
  - Goes beyond just NFR and SLA and depends on use case
  - Personalisation is not just a question of the interface and the data, but of the whole software behaviour stack
  - Interoperability and portability cannot be posed on the developer

  ➔ *incentive of the system*

consequences / approach

TOWARDS I3

an increased abstraction *without performance degradation* is necessary to improve interoperability, usability, portability …

requires strong rethinking of traditional methods

- **The complexity of software increases so much that maintenance and adoption become near impossible**
  - New abstractions are needed that encode the *intention* of the program, much more than its behaviour
  - Must give enough freedom for the execution environment to adapt the actual code
  - Must be close to the hardware, not adding another abstraction layer leading to incompatibilities

➔ *intention of the system*

consequences / approach

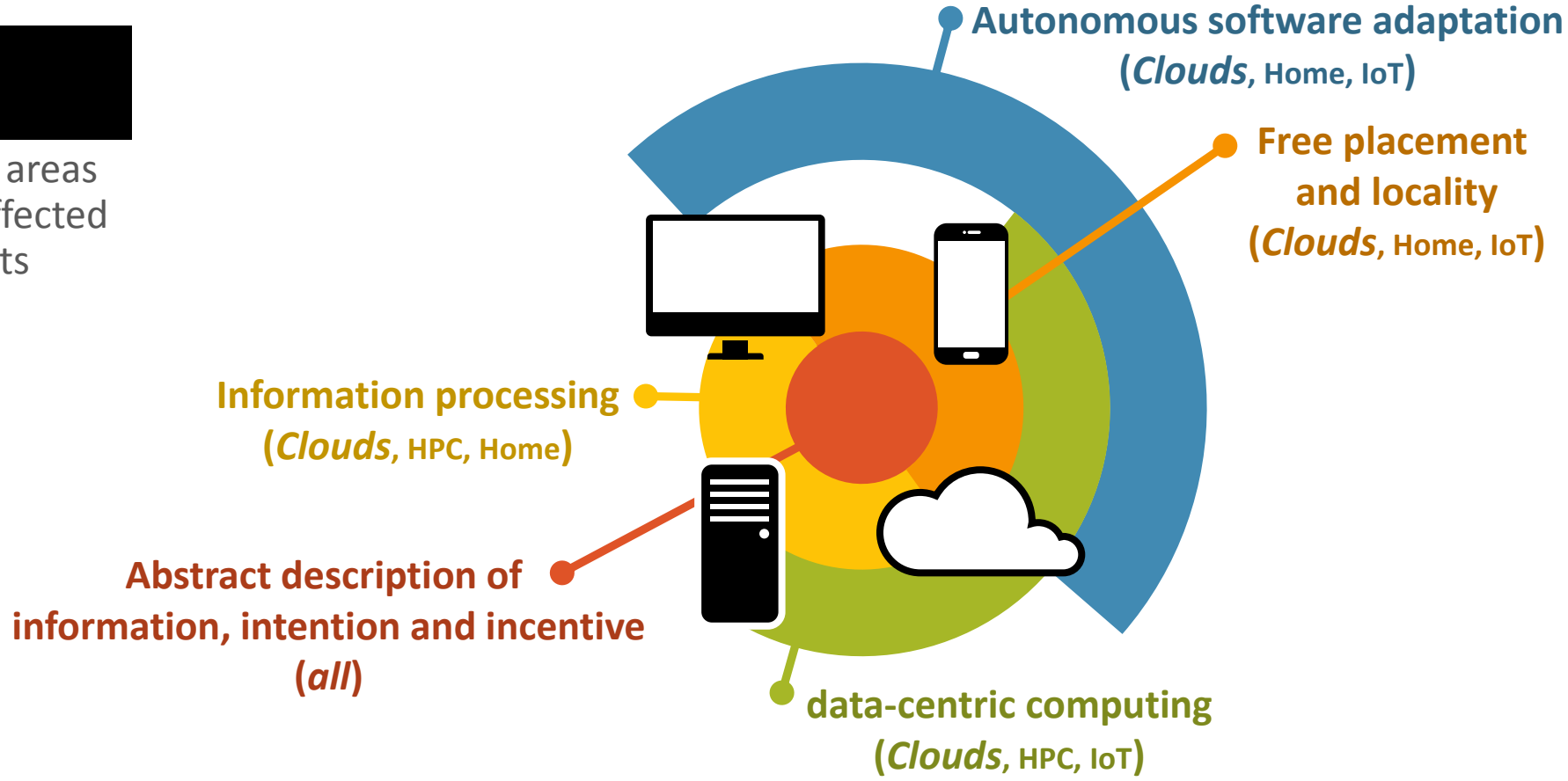
I3

- **Information:** IT is about information and knowledge. Interoperability by abstracting from data and maintaining information
- **Incentive:** Software needs to meet the functional and non-functional requirements, but they need to be “naturally” and understandably defined
- **Intention:** Every software has a goal, the algorithm is just one path towards that

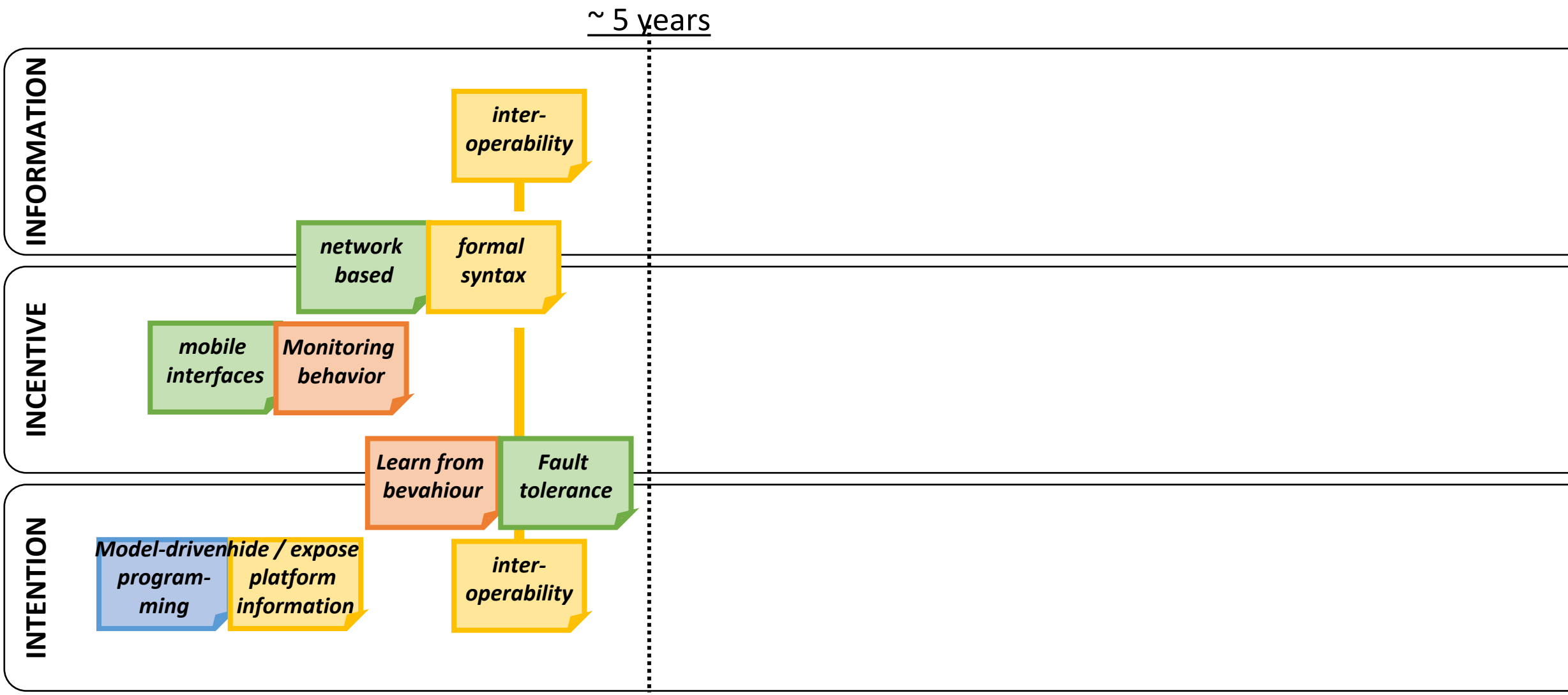
putting it in context

AREAS OF RELEVANCE

different market areas are differently affected by the key aspects

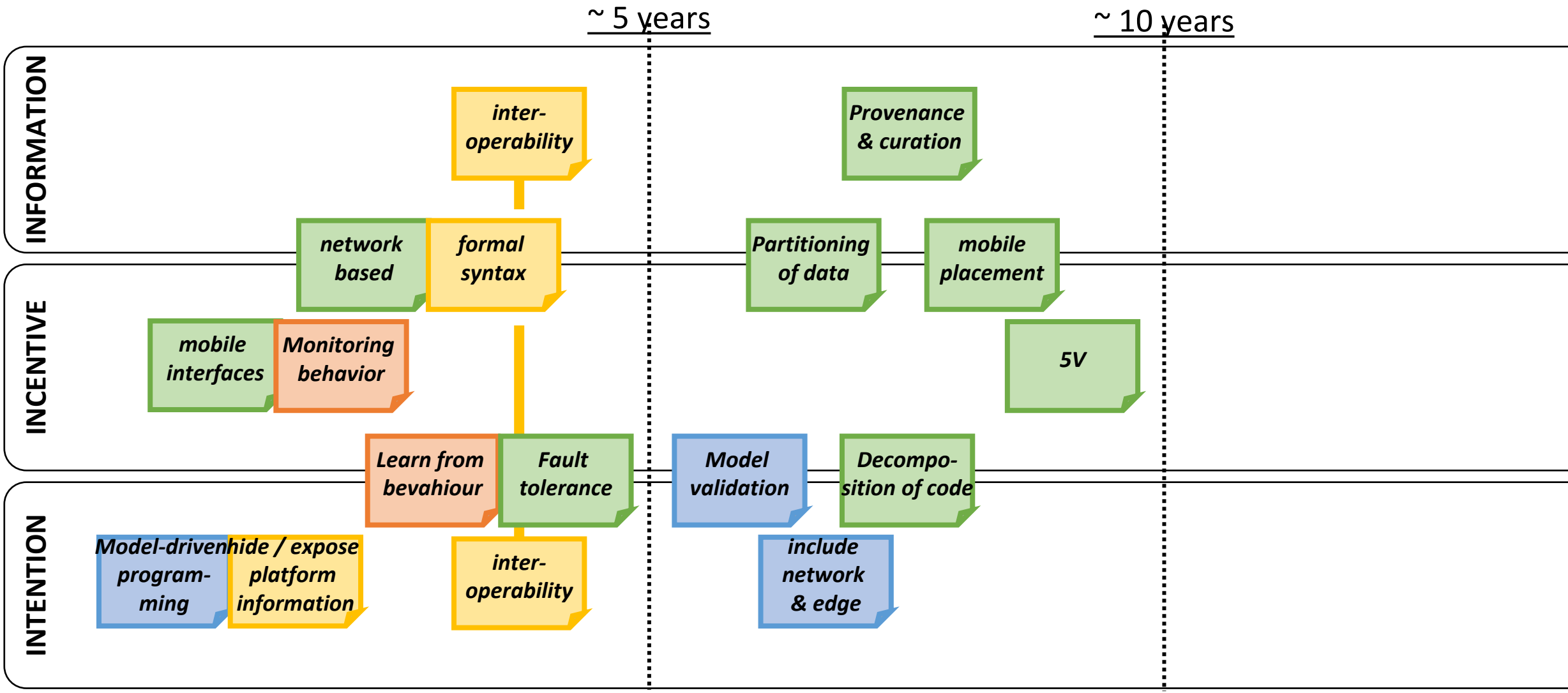


the timeline

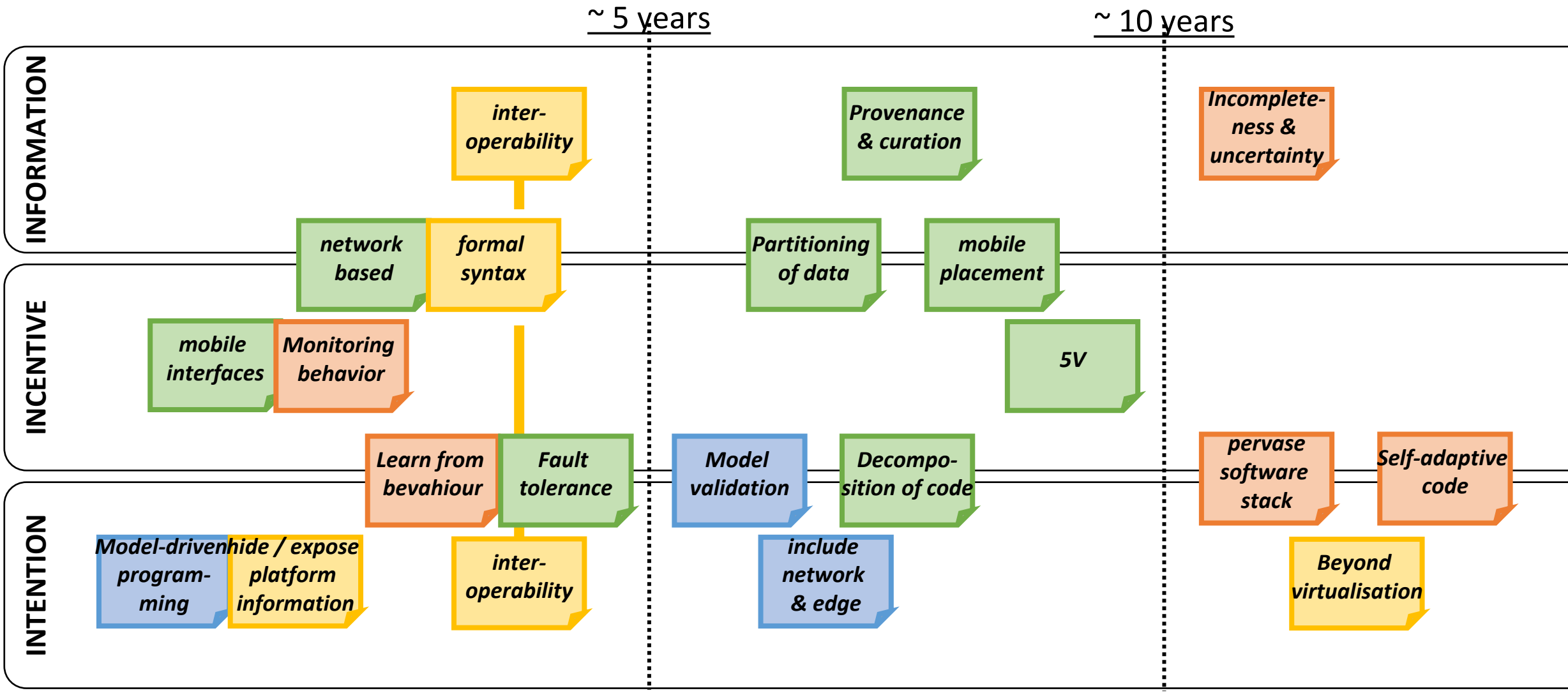


Interoperability & Portability Placement & Locality Advanced System Development Adaptive Code

the timeline

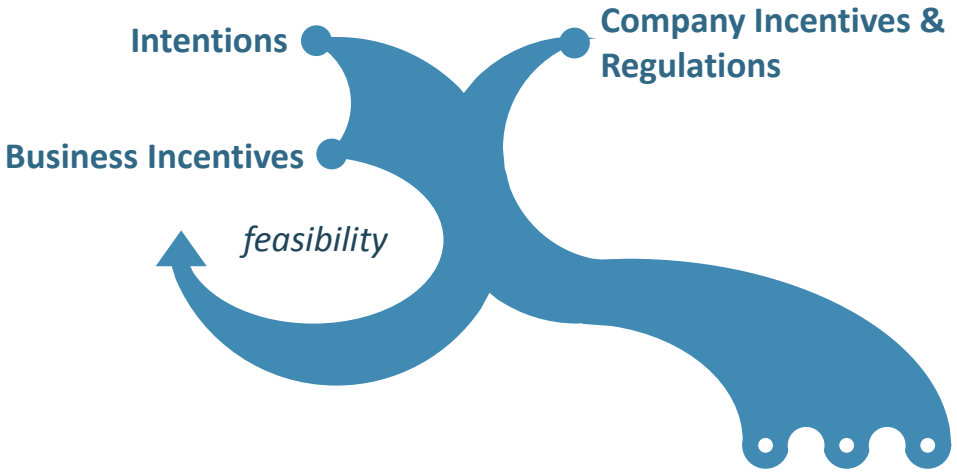


the timeline



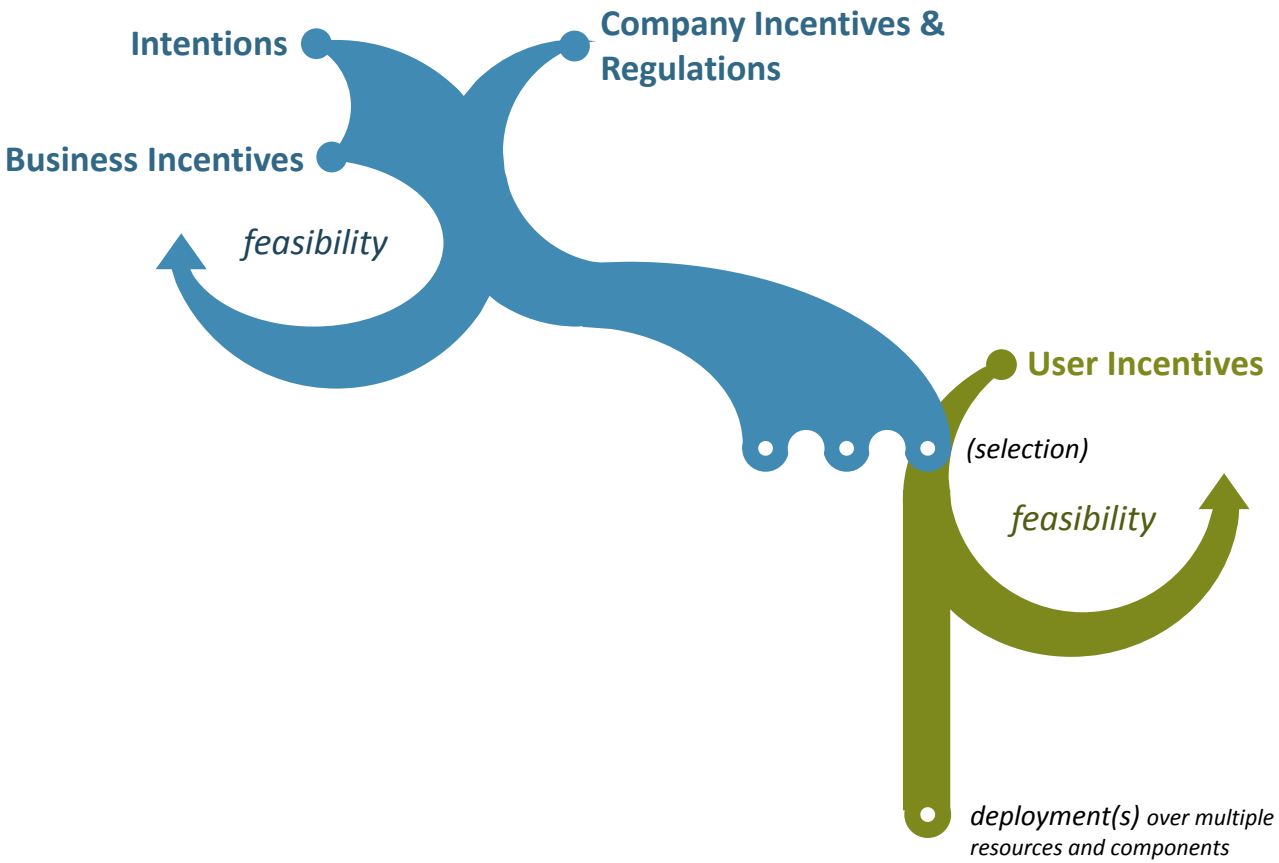
… putting it into context

"what car design would optimally meet our typical customers' needs while maintaining our company incentives?"



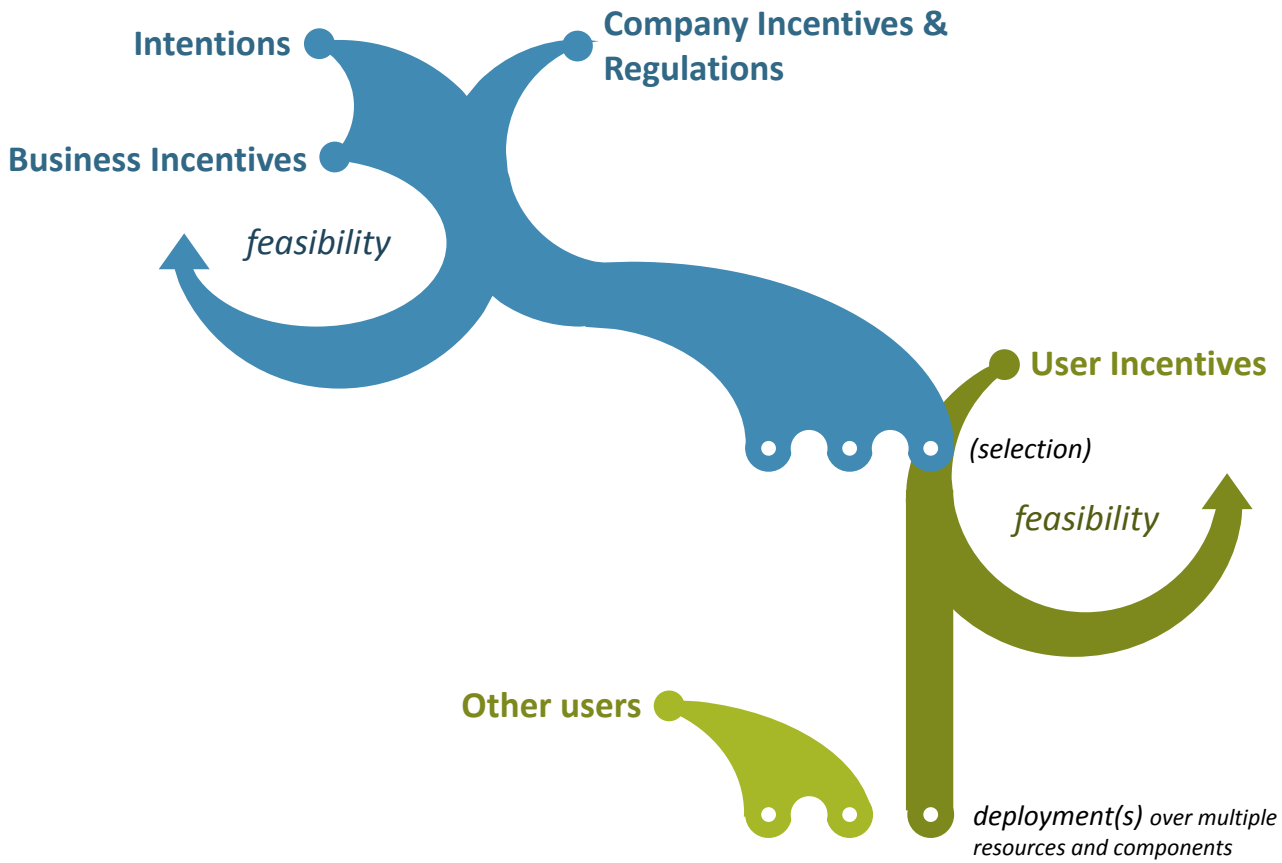
… putting it into context

"what car design would optimally meet our typical customers' needs while maintaining our company incentives?"



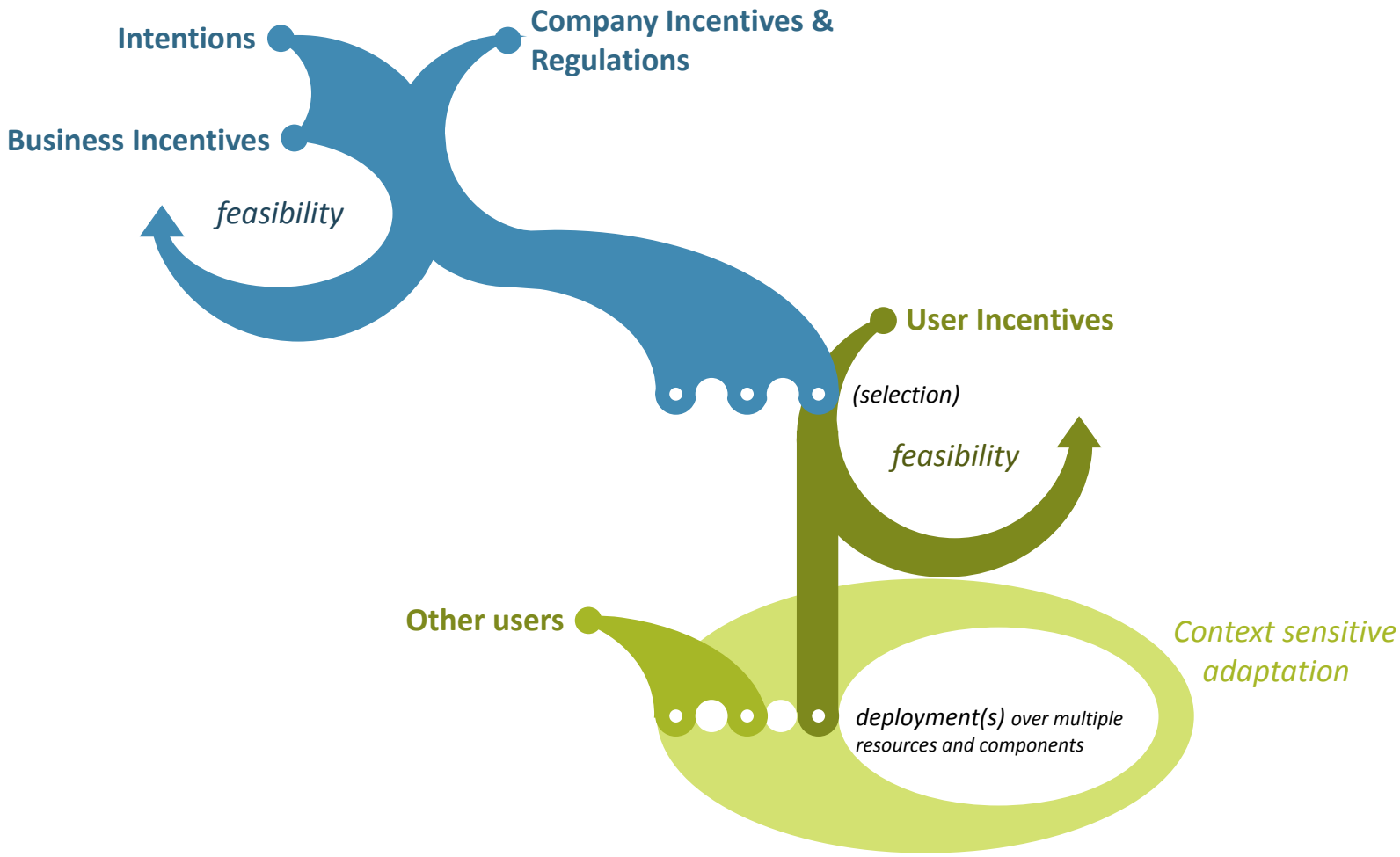
… putting it into context

"what car design would optimally meet our typical customers' needs while maintaining our company incentives?"



… putting it into context

"what car design would optimally meet our typical customers' needs while maintaining our company incentives?"



More details:

Keith Jeffery, Lutz Schubert et al., “Beyond Cloud Computing: towards Complete Computing - the CF2016 roadmap”, to be published

Lutz Schubert, Keith Jeffery, "New Software Engineering Requirements in Clouds and Large-Scale Systems", IEEE Cloud Computing vol. 2 no. 1, p. 48-58, Jan.-Feb., 2015

Keith Jeffery, Lutz Schubert, “Complete Computing: toward information, incentive and intention”, EC report. Available at: http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?action=display&doc_id=6775