IN5320 - Development in Platform Ecosystems

Lecture 9: *Design in platform ecosystems*

15th of October 2018

Department of Informatics, University of Oslo
Magnus Li - magl@ifi.uio.no

Today

Two problems in large-scale / generic software development:

- "Generic" usability
- Working with local users in development

We will look at:

1. Usability.
2. User participation in design.
3. Our two problems.
4. Participation and scale. Four types of participatory design.
5. Architectures for participation and local adaptations.
6. IN5320 individual assignment award 2018!

Usability and participation

Usability

How well a system works for the user

Do the system allow the intended users to certain goals with

- Effectiveness (doing the right things)
- Efficiency (doing things right)
- Satisfaction

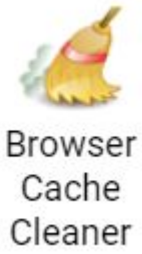
Usability - Nielsen’s 10 heuristics

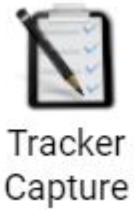
- Visibility of system status
- **Match between system and the real world**
- User control and freedom
- Consistency and standards
- Error prevention
- **Recognition rather than recall**
- Flexibility and efficiency of use
- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Help and documentation

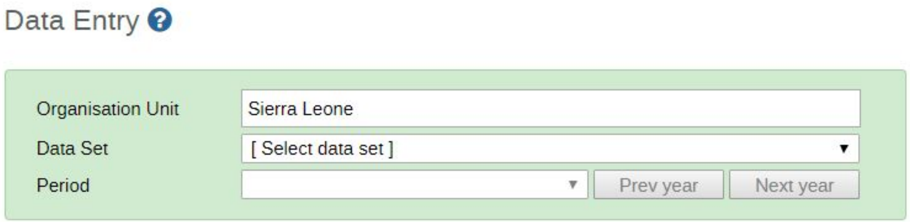
Different mental models

System oriented versus real-world oriented language

DHIS2 concepts



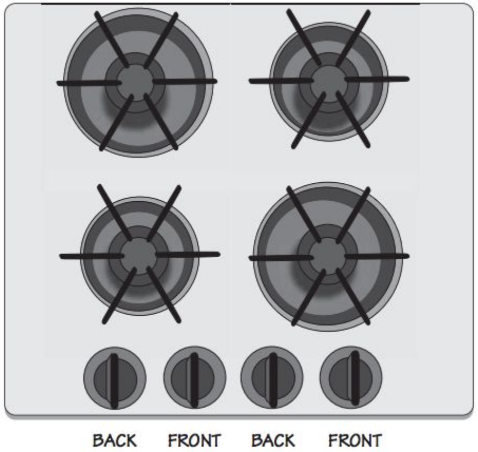




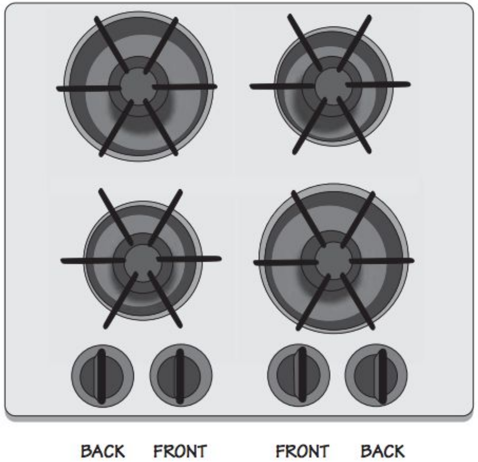
Usability - Donald Normans 6 principles

- Affordances and signifiers
- Mapping
- Consistency
- Constraints
- Feedback
- Visibility

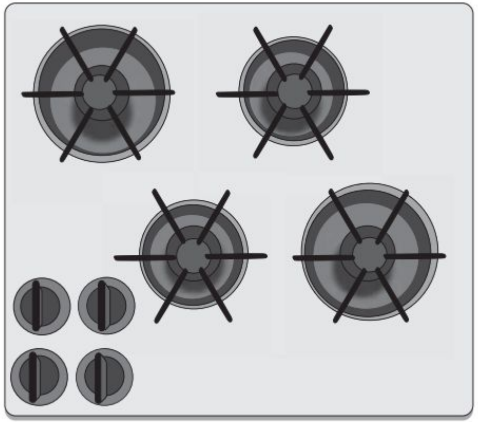
A.



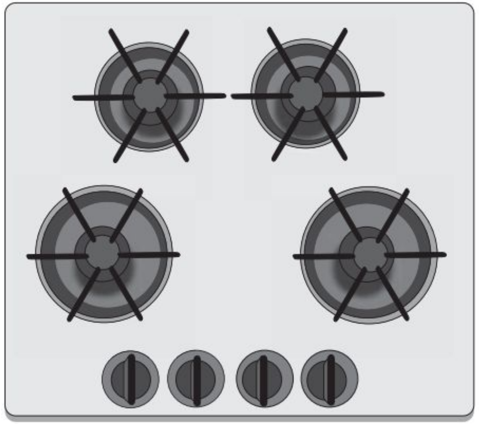
B.



C.



D.



Norman 2013

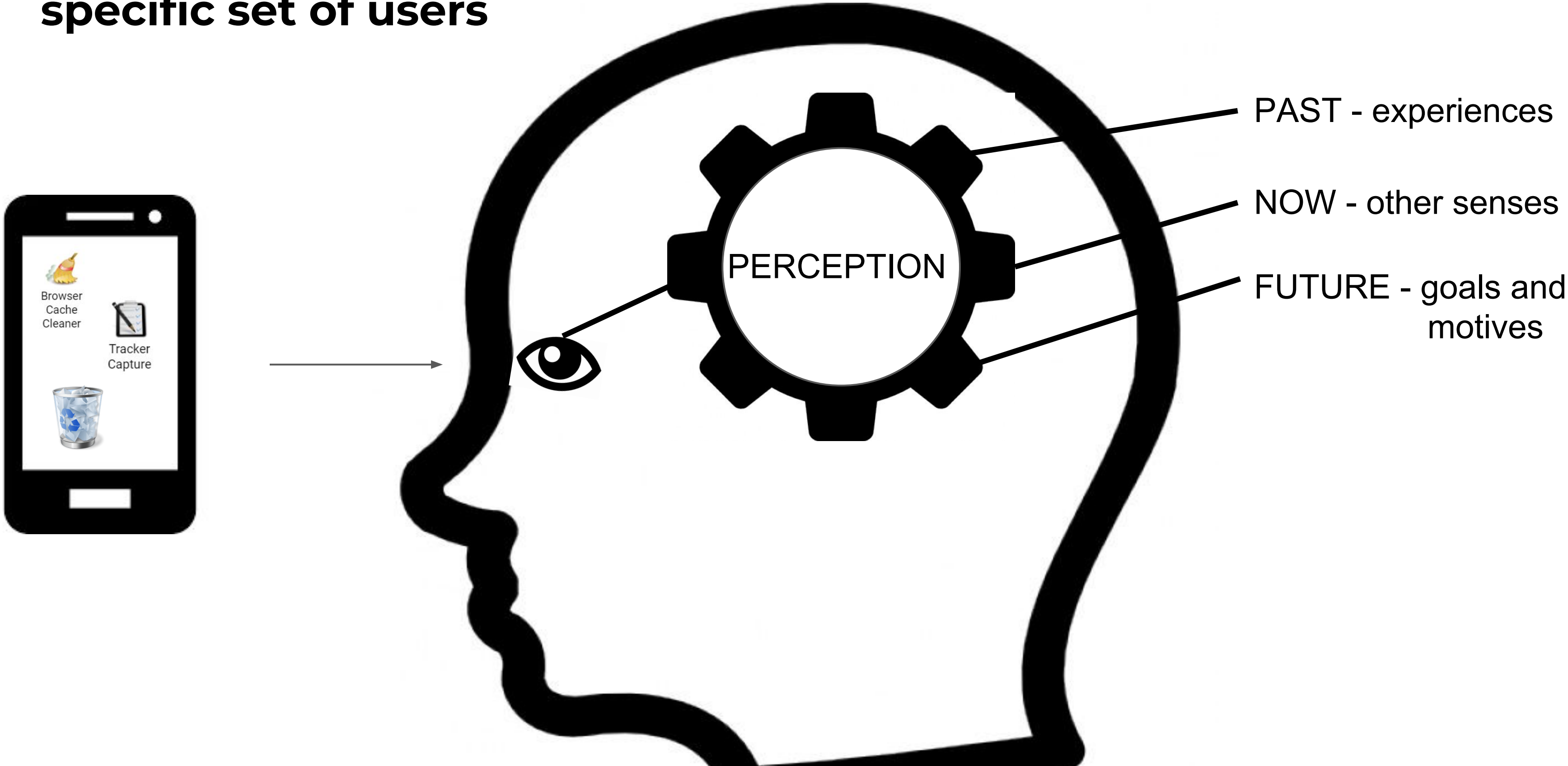
3 hrs ·

Hva betyr det da den gensern der bare står å blinker å vaskemaskina gir faen i alt ? Får ikke opp døra eller noe..


PANIKK
LYS 1
OPP
DIAS
NED
STRØM
SKINNE
2
Projektor
strøm
bak
INGEN
FUNKSJON
OPP
PÅ/AV
VIDEO
PROJEKTOR
PROJEKSJONS
LERRET
TUNGHØRT
ANLEGG
HØYTTALER
ANLEGG
TALE
VOLUM
NED

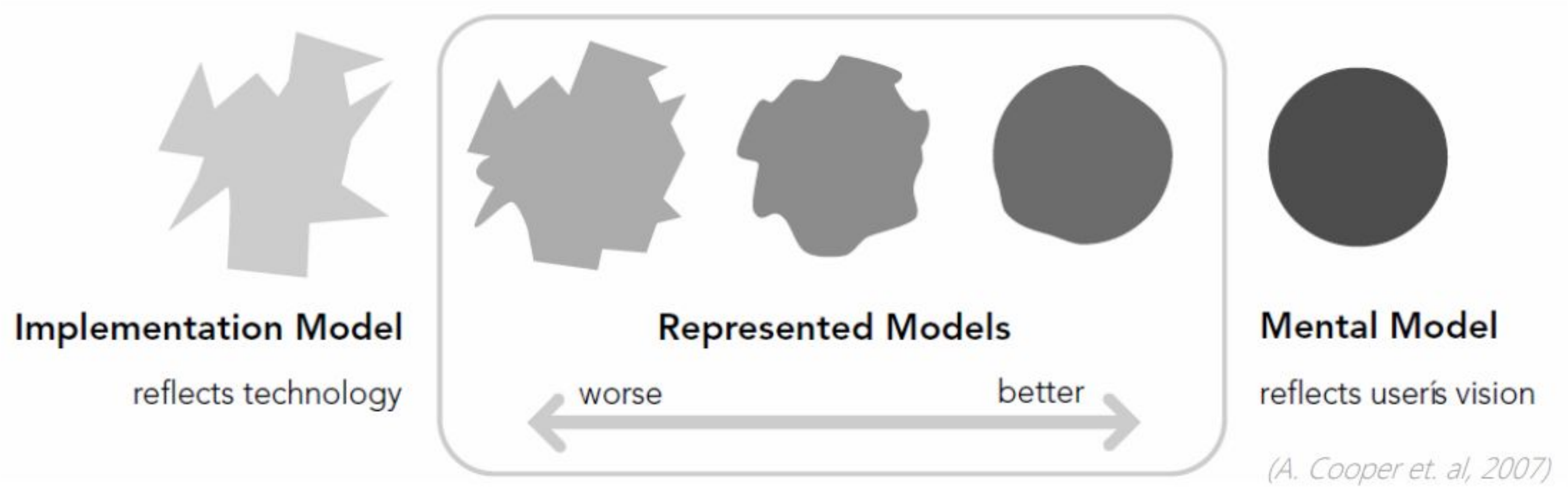
Usability

Not just defined by qualities of the software, but dependent on a **specific set of users**



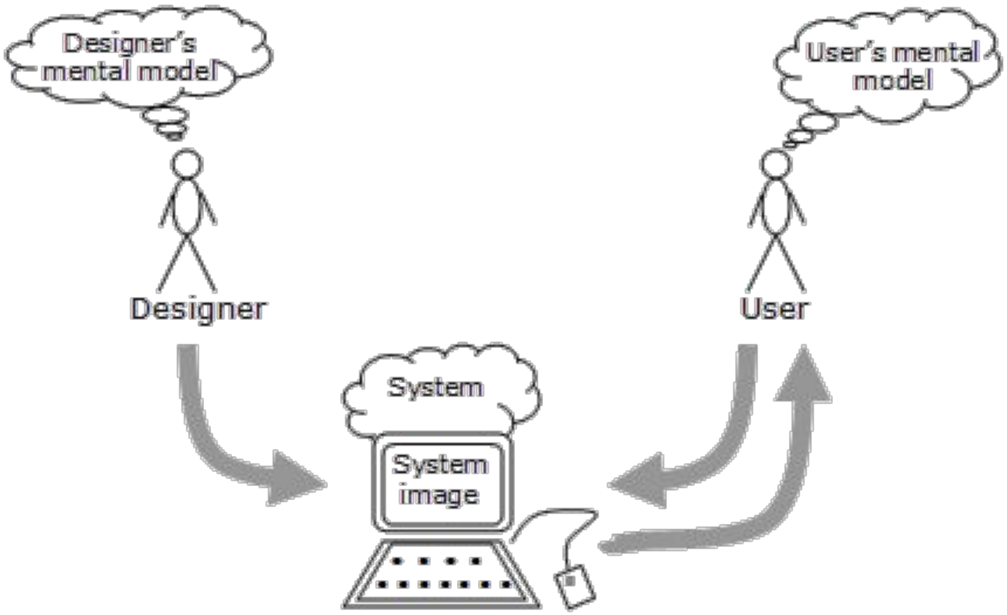
Mental models

To make systems intuitive and usable, designers must try to create interfaces between technology and the user that are close to their mental model.

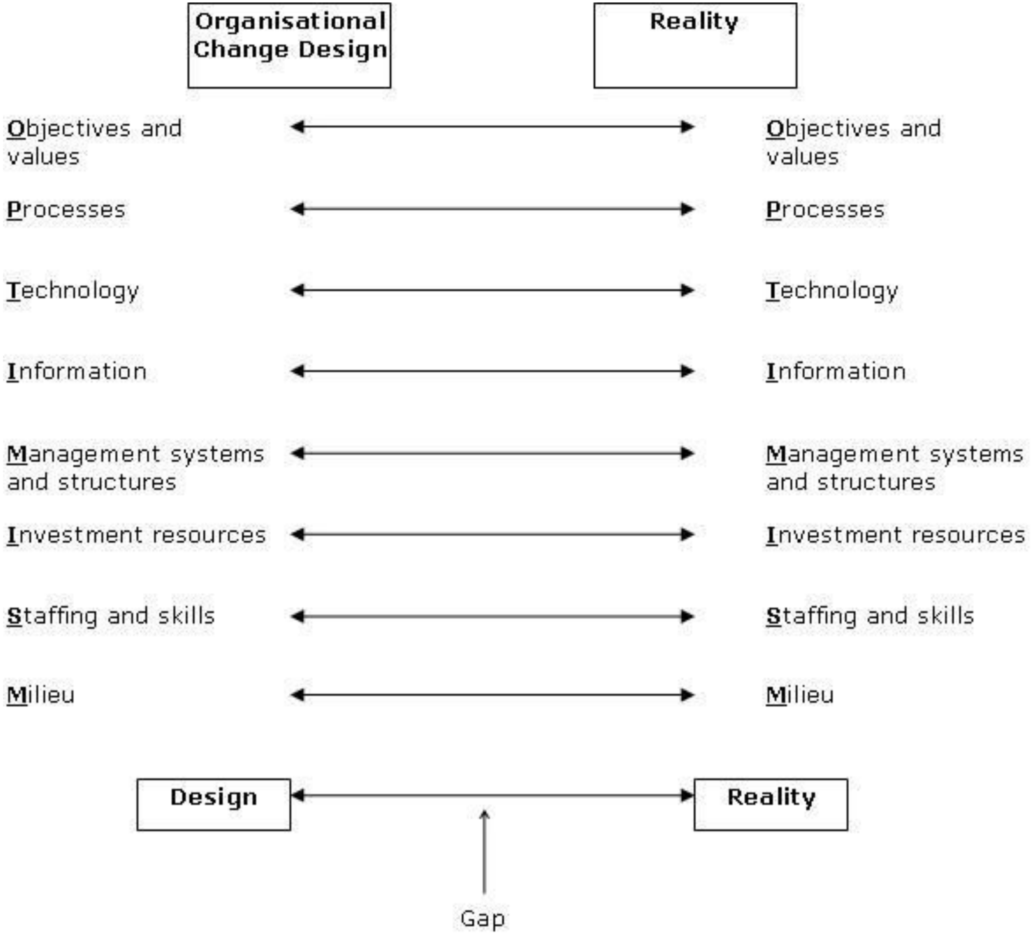


Design - actuality gaps

Developers, designers, and end-users may have radically different understandings of the world.

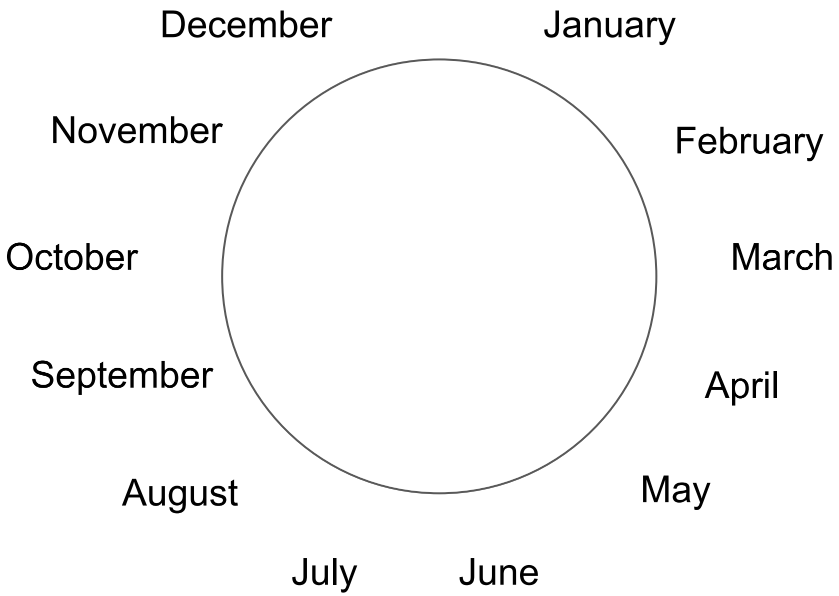


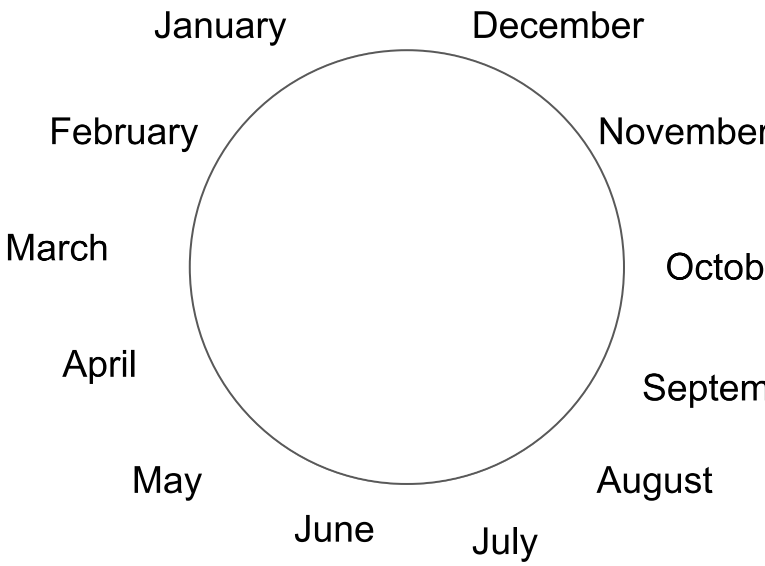
Design - actuality gaps



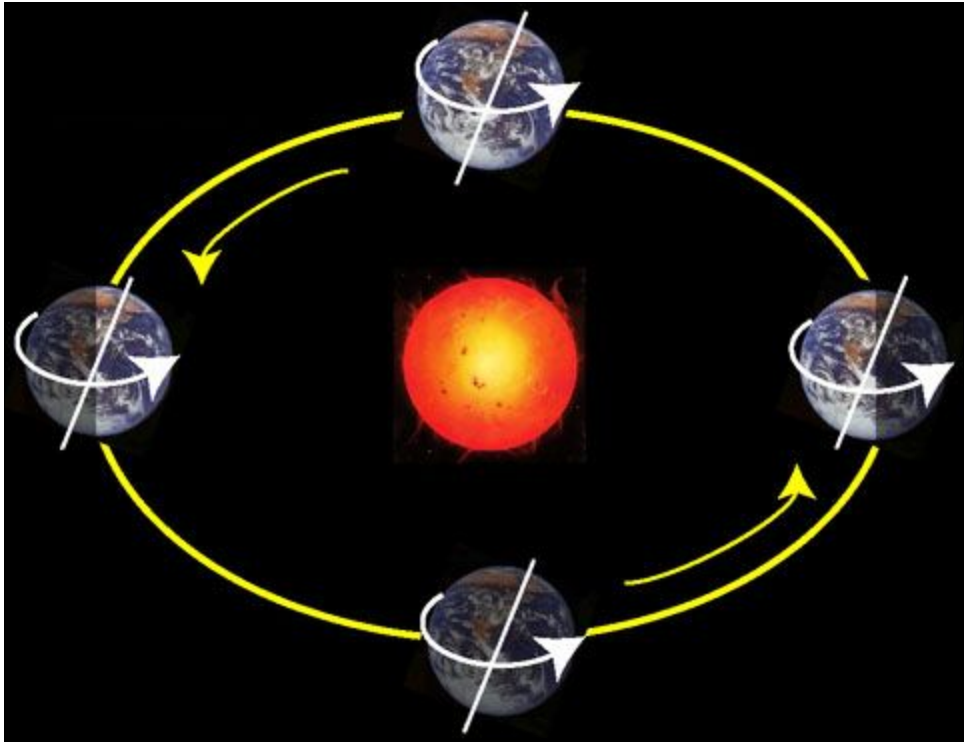
Heeks 2002

Different mental models





Different mental models



Usability

Not just defined by qualities of the software, but dependent on a specific set of users **in a certain context of use.**

User participation in design

To ensure usable systems, we must understand the users and context that we design for.

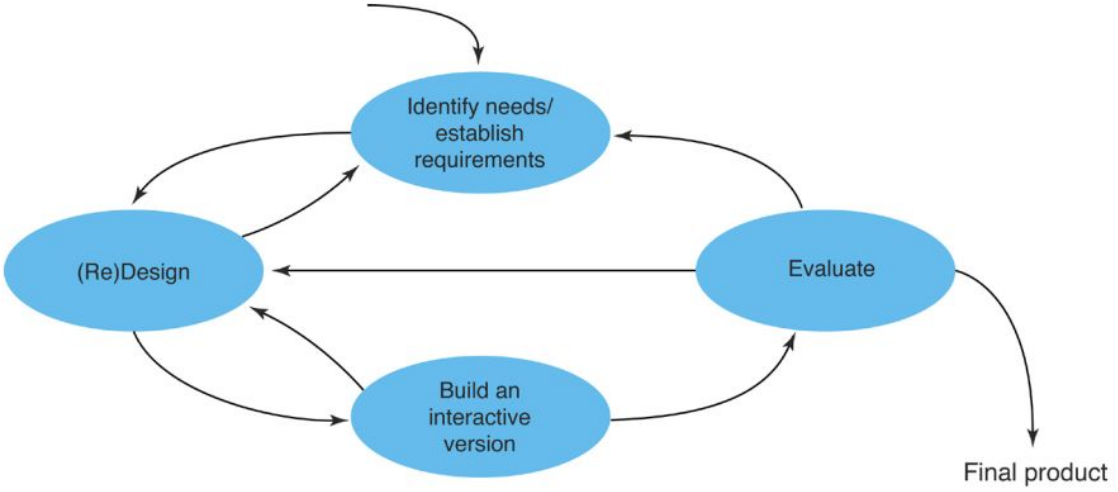
A common way to do this is to:

1. Investigate the context
2. Involve end-users in the development process.

User participation in design

Several traditions. Two common are.

- User/human-centered design (UCD).
- Participatory design.



UCD-model

User participation in design

Different levels of participation.

Informative

Users provide and/or receive information

Consultative

Users comment on a predefined service or range of facilities

Participative

Users influence decisions relating to the whole system

Figure 3-2 Forms of user involvement (Damodaran, 1996, p. 365)

Challenge of scale

Challenges in large-scale systems development

Two problems in large-scale / generic software development:

1. “Generic” usability

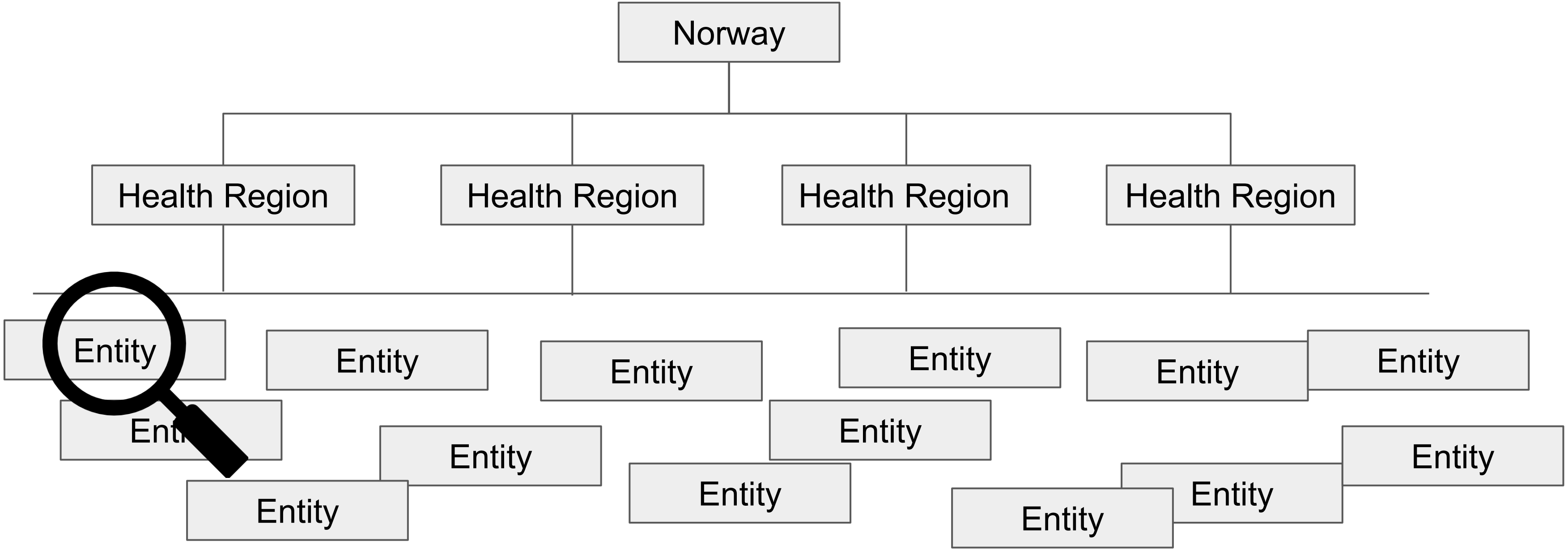
2. Working with local users in development

What if we are developing software to be used by very different users in very different contexts?

Challenges of large-scale systems development

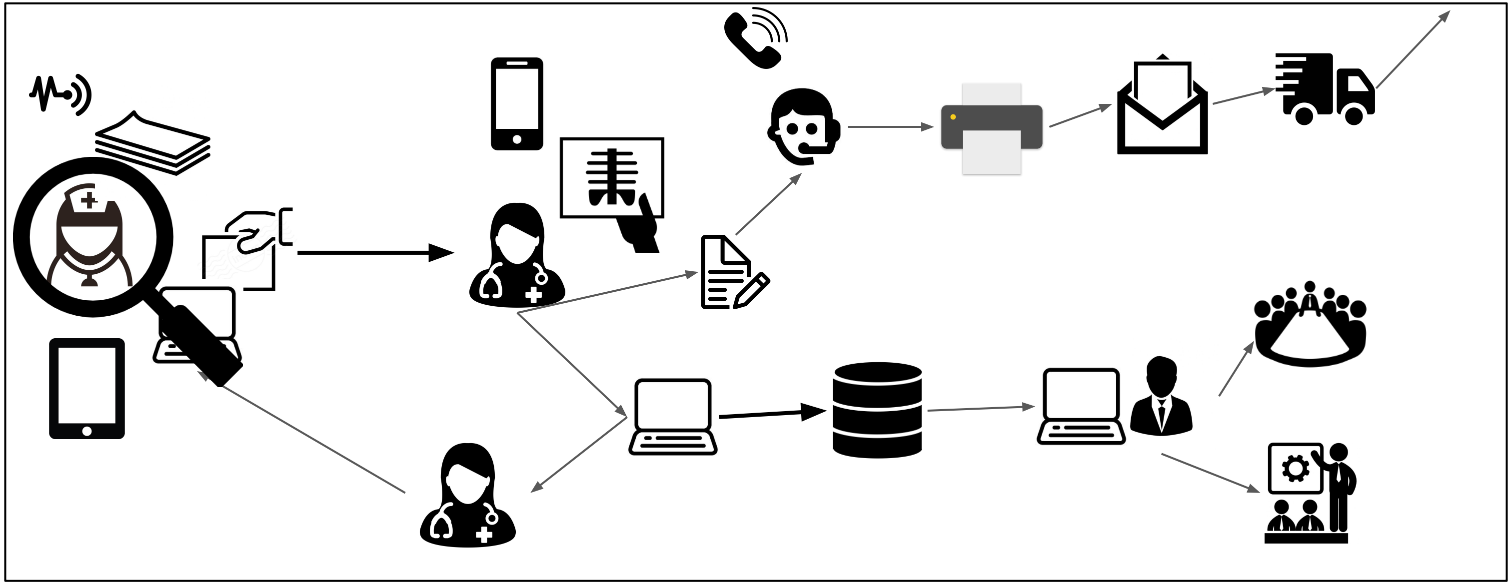
Example 1 - Standardized Patient Journal system

Implementing one common patient journal across different departments, hospitals, regions.



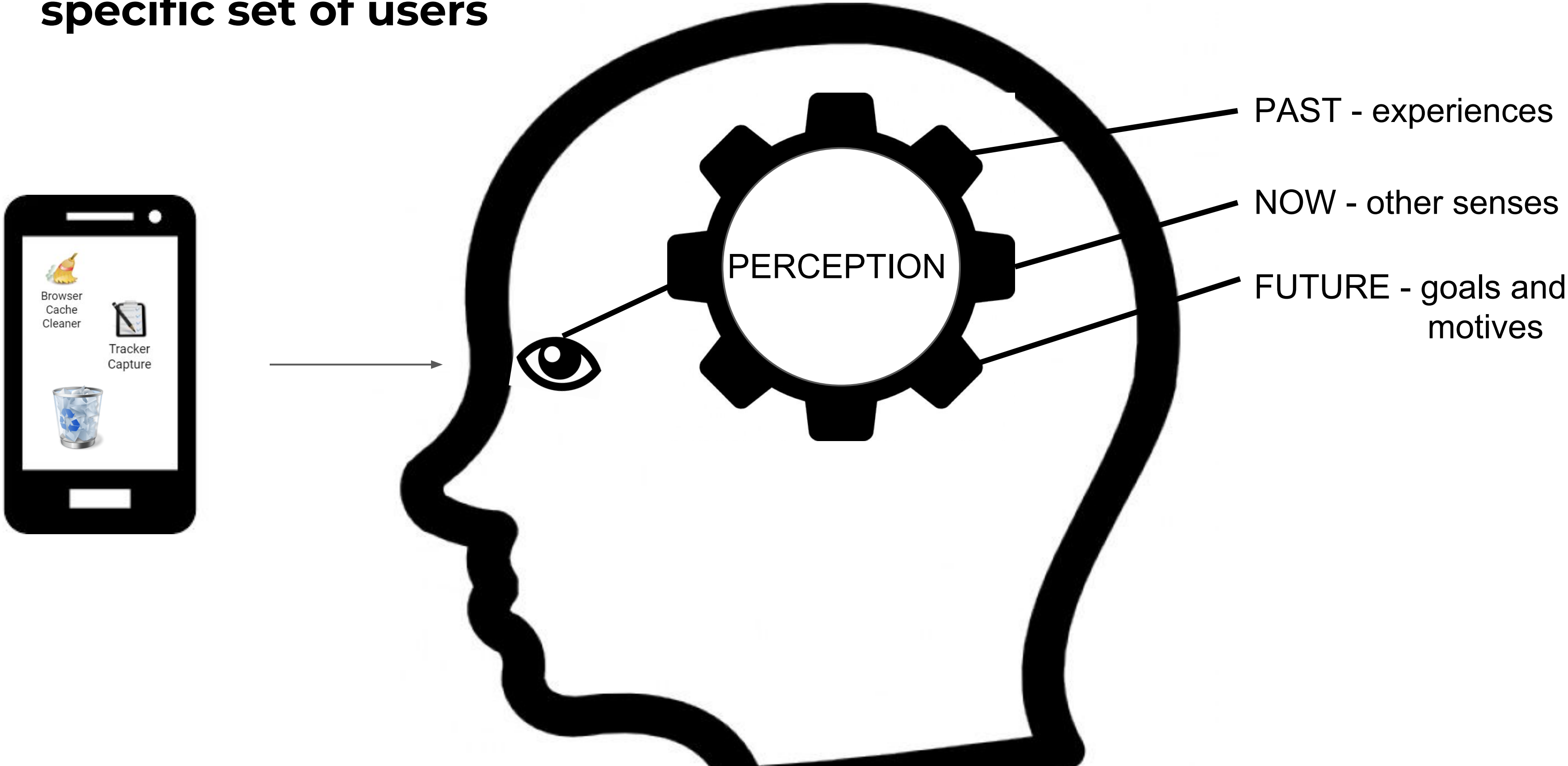
Challenges of large-scale systems development

Work practices, routines, language / semantics, culture, norms, legacy systems, dependencies etc.



Challenges of large-scale systems development

Not just defined by qualities of the software, but dependent on a **specific set of users**



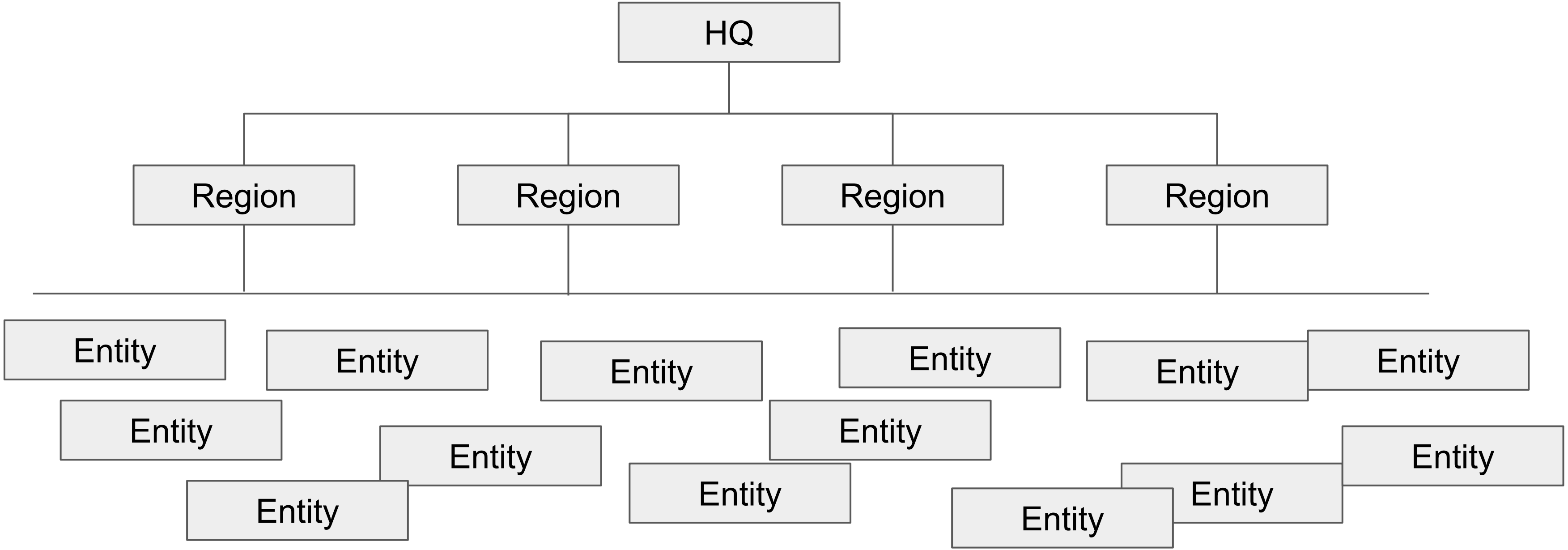
Challenges in large-scale systems development

- Usability is dependent on the users
- Who is it usable for?
- Who to involve in design?

Challenges of large-scale systems development

Example 2 (Rolland & Monteiro, 2002) - Large shipping survey company:

Implementing one common system to support surveying across 300 sites in over 100 countries.

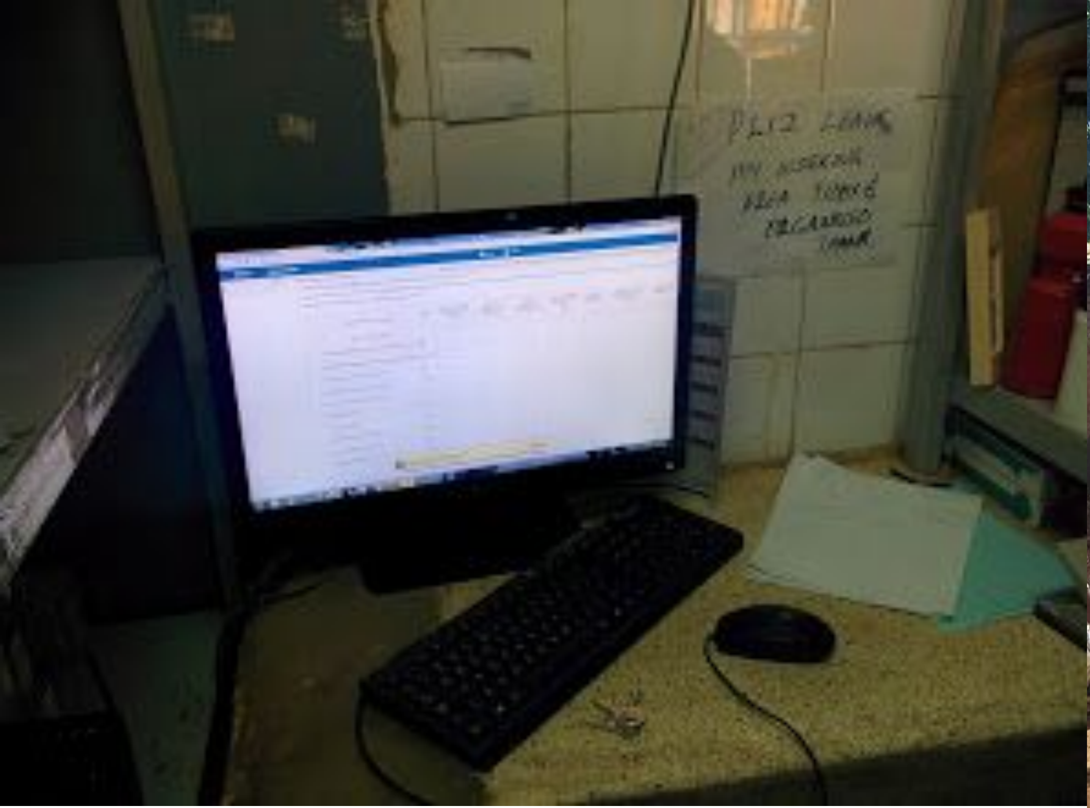


Challenges of large-scale systems development

Example 4 - HISP / DHIS2

Timeline	*Stage*	*Use and Development*
1994-1999	Pilot and national system	Users and collocated software developers, all in South Africa, network of users
2000-2004	Expansion	Multiple countries, core development isolated from local modifications
2004-2007	Technological transition	Two branches (v1 & v2), infrastructure for sharing, but fragmented processes, isolated modifications
2008->	Integration	Multiple local teams, travelling, local developments contributing to global software

Titlestad et al., 2009

Forms of participatory design (Roland et al., 2018)

Rolan et al,. 2018 have identified four types of participatory design based on scale (number of heterogeneous users and settings)

Singluar PD

Serial PD

Parallel PD

Community PD

Singular PD

Singular - classic participatory design

Design technology in cooperation with small group of end-users.

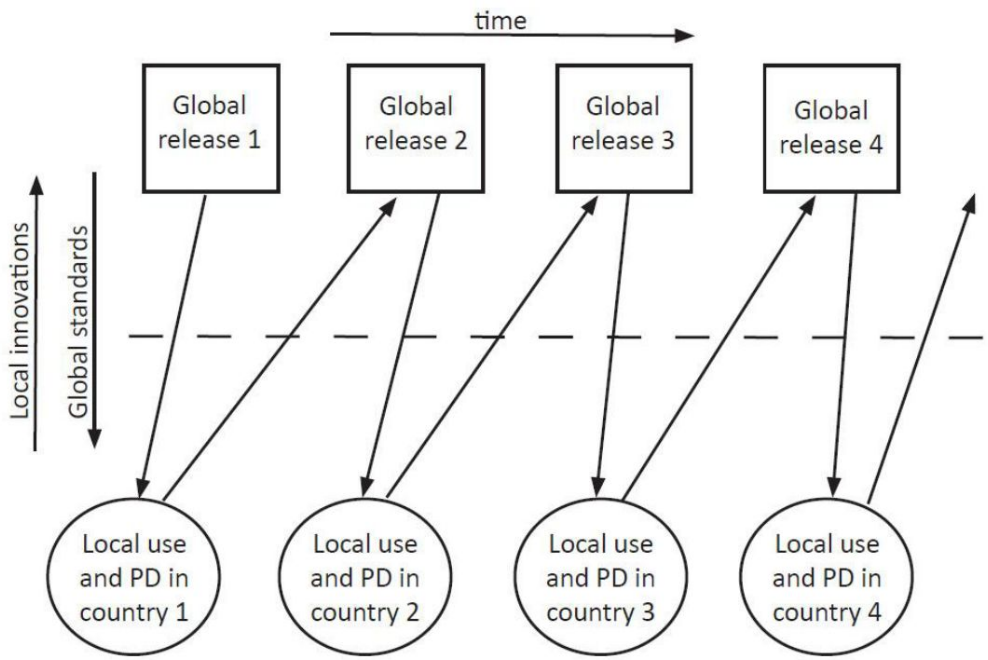
Mutual learning

End-users can take part in fundamental decisions

Serial PD

Design of artifact used in multiple settings / organizations / groups of users

In cooperation with end-users at one site, then another, and so forth.

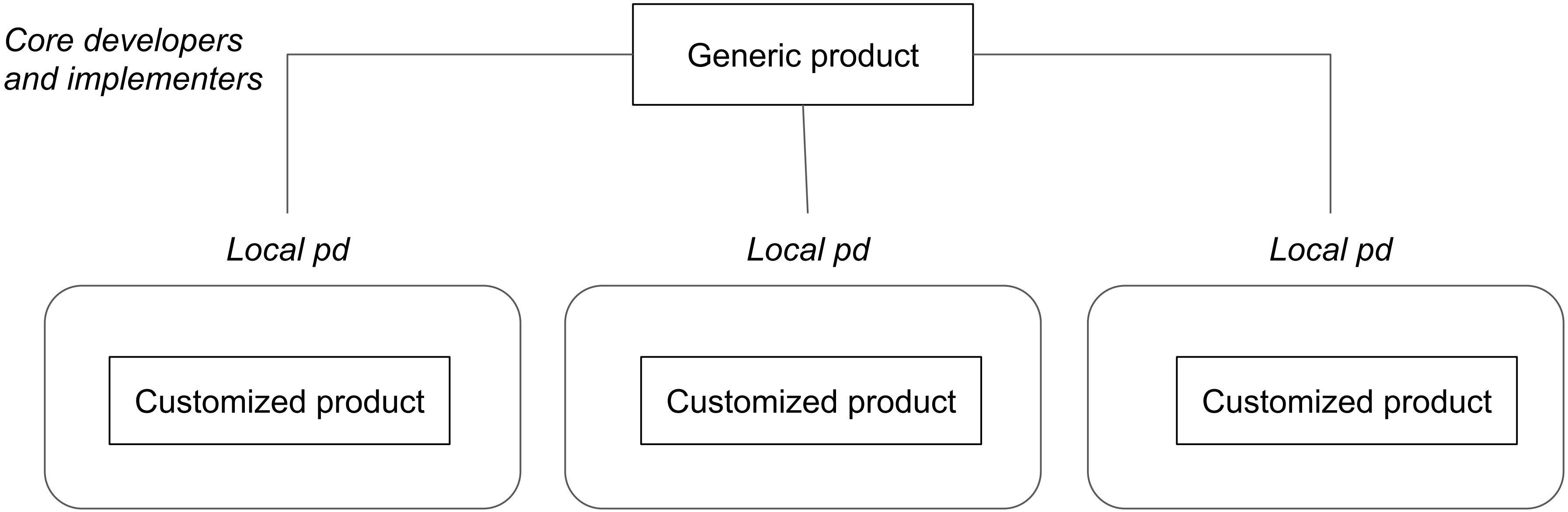


Titlestad et al., 2009

Parallel PD

Users are engaged at several sites in parallel to inform *generic* design

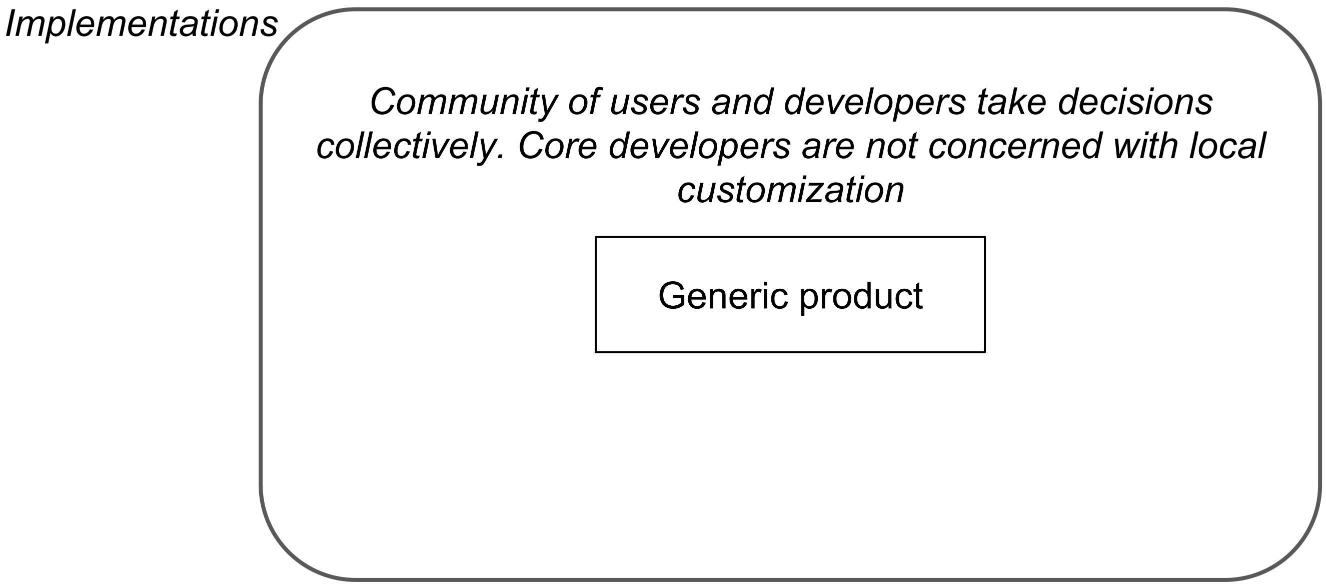
Core developers make visits to sites in parallel.



Community PD

Broader community negotiates generic features. Local customization without involvement from core developers.

Circulating use-cases and best practices. Workshops and online arenas for communication



Meta-design and platforms

Architectures

- The technology must allow for local adaptations
- Flexibility for customization
- Modularization to innovate and make local adaptations.
- Open source software

Meta-design

Meta-design: Designing for future design (Andries Van Onck, Gerard Fischer e.g., 2008).

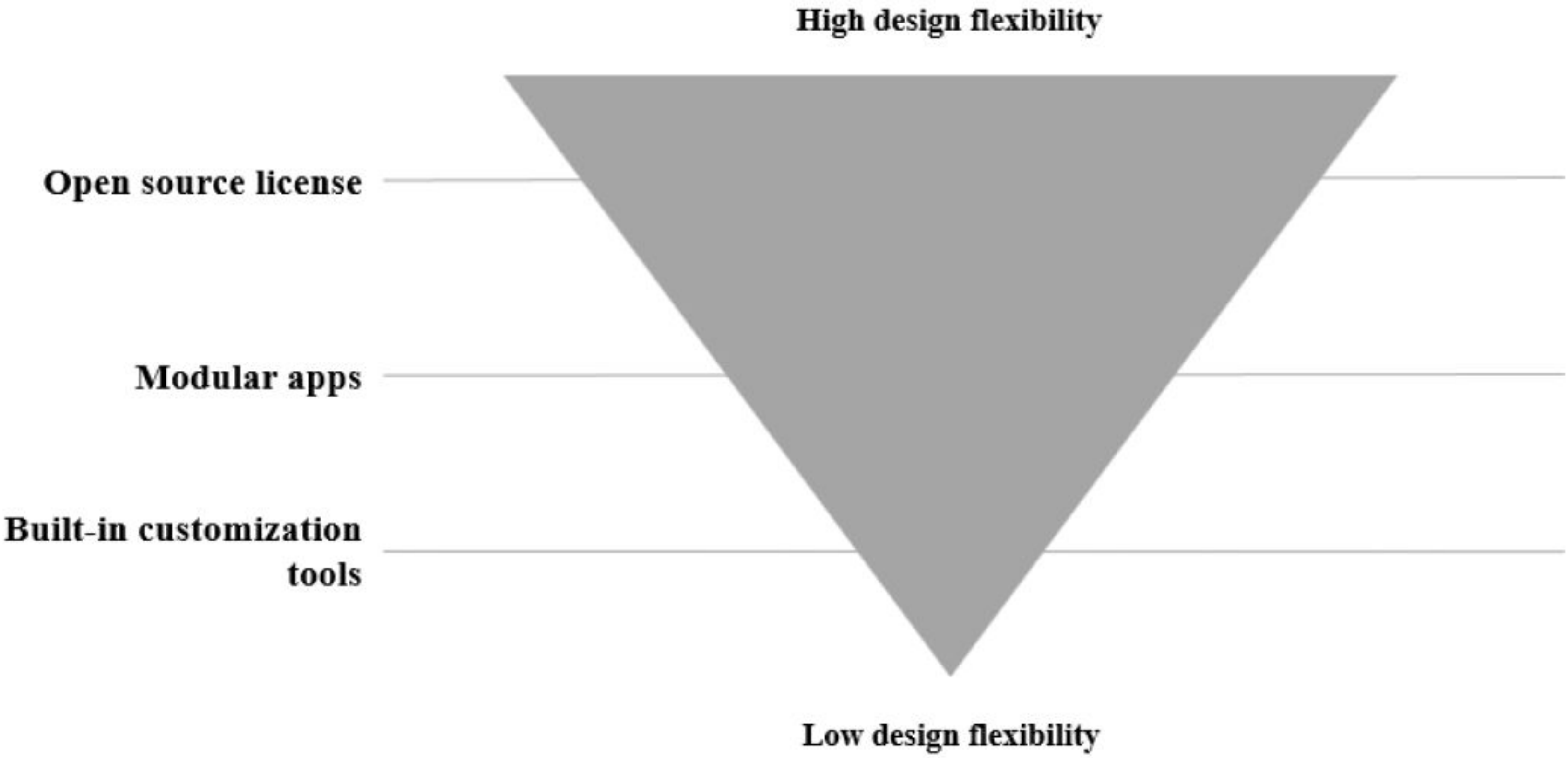
- Design continue during use.

Software developers create “spaces” so that the software can be shaped according to local use during or after implementation.

- Making functionality and interfaces customizable
- Enable development of plugins
- Open source software.

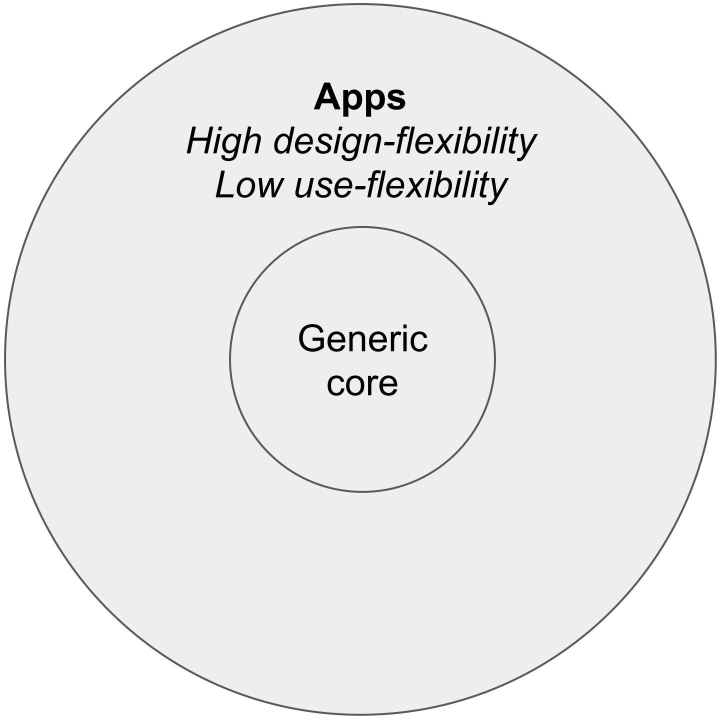
→ Mainly aimed at end-users as designers.

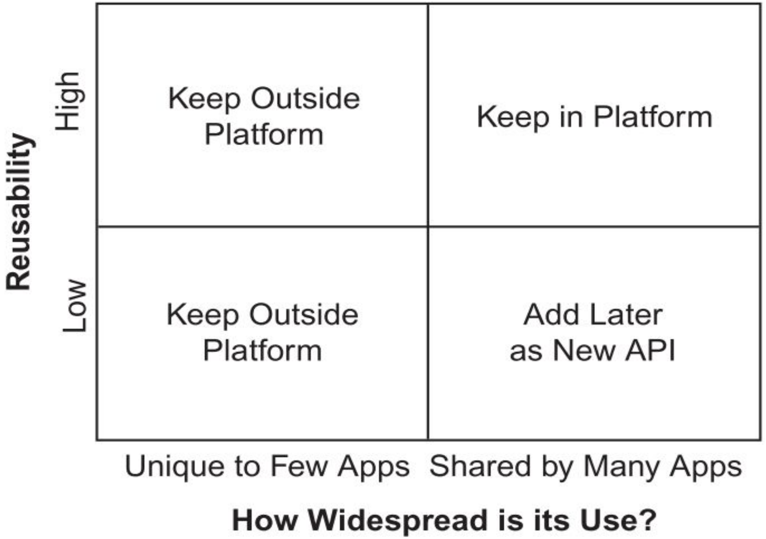
Meta-design


High design flexibility
Open source license
Modular apps
Built-in customization tools
Low design flexibility

Platform design flexibility

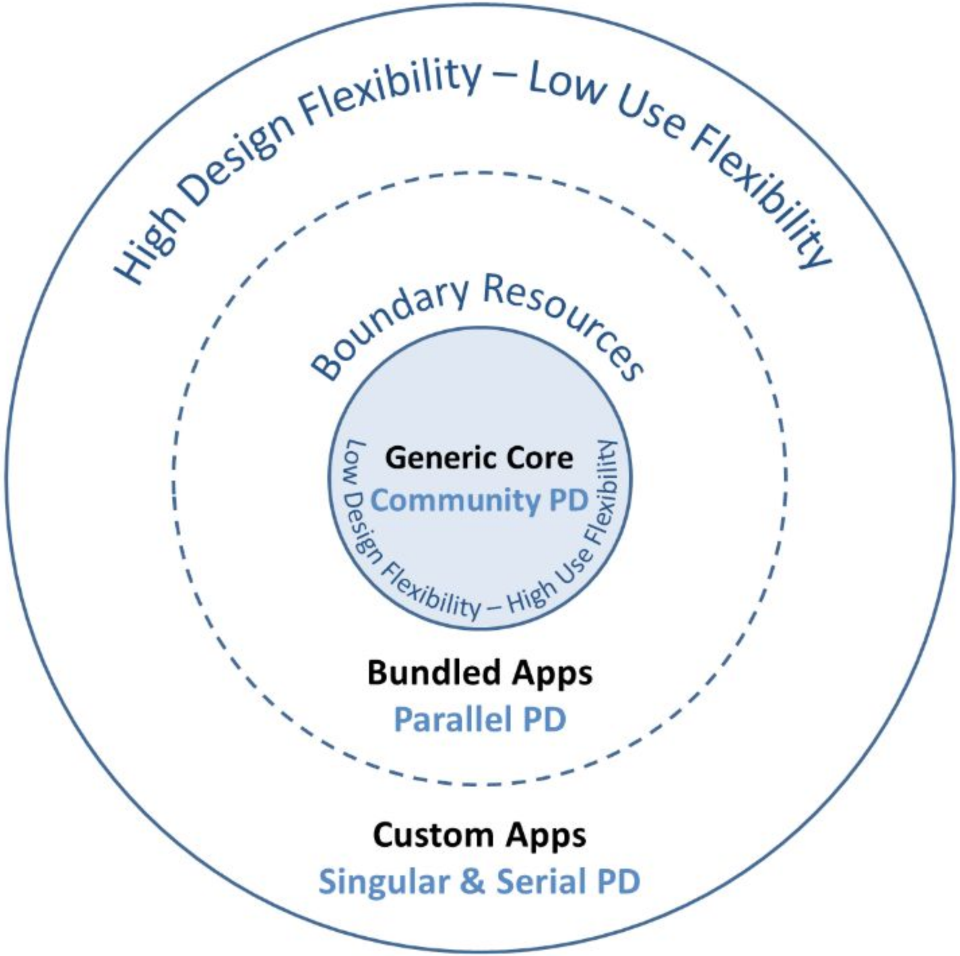
Opening up the software architecture for the development of third-party apps could be one way of providing local implementers with flexibility.





Tiwana 2013

Platform architectures to support PD



Roland et al., 2018

Architectures

- Technology is not enough.
- Also a “social architecture” is needed.

Need of

- Local competence
- Channels of communication

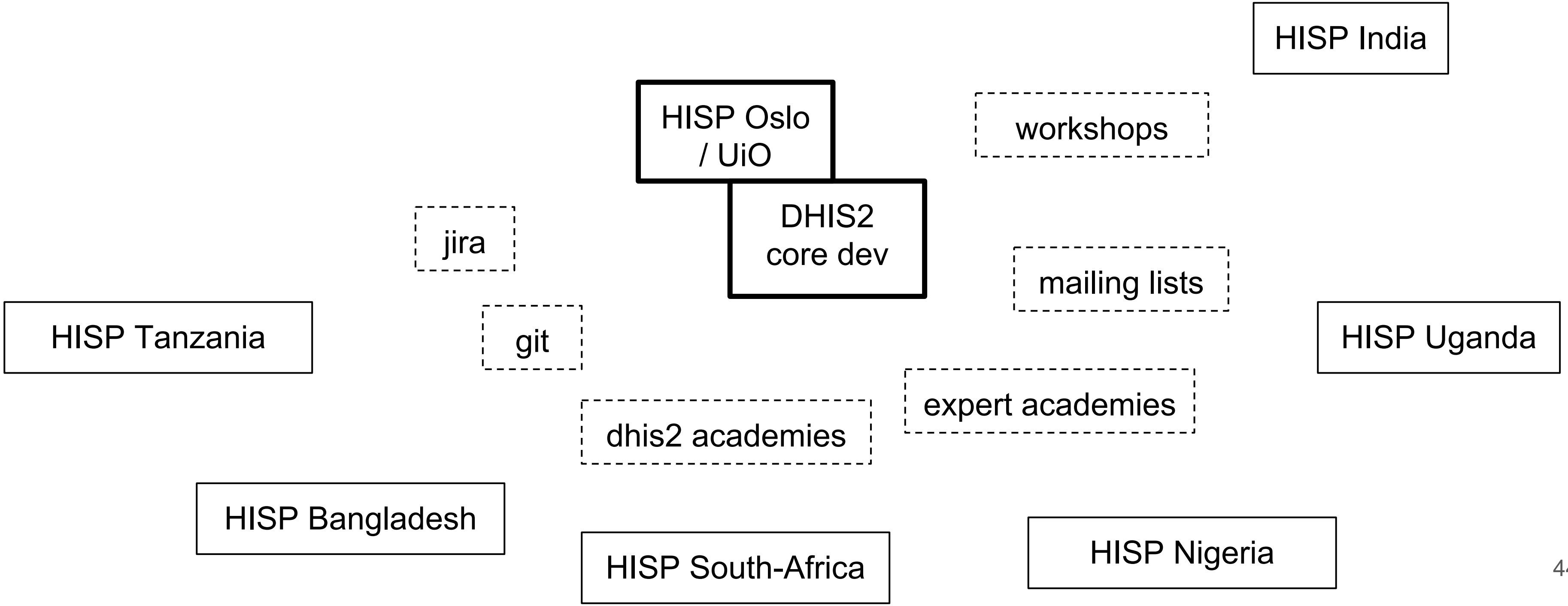
Enabling large-scale distributed design

Scaffolding - structure that supports design and implementation (Titlestad et al., 2009)

"for the duration of a particular human practice, actors draw on various artefacts, spaces, and infrastructures to conduct their activities" - Orlikowski 2006 p462

Enabling large-scale distributed design

Scaffolding - structure that supports design and implementation



Enabling large-scale distributed design

Boundary spanners

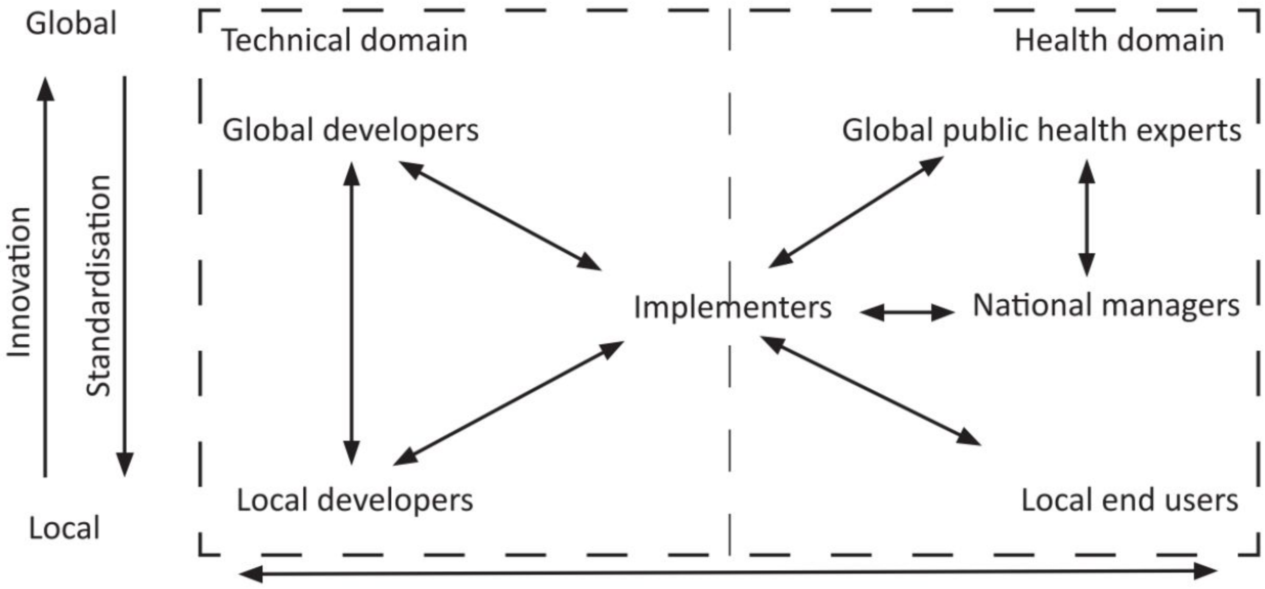
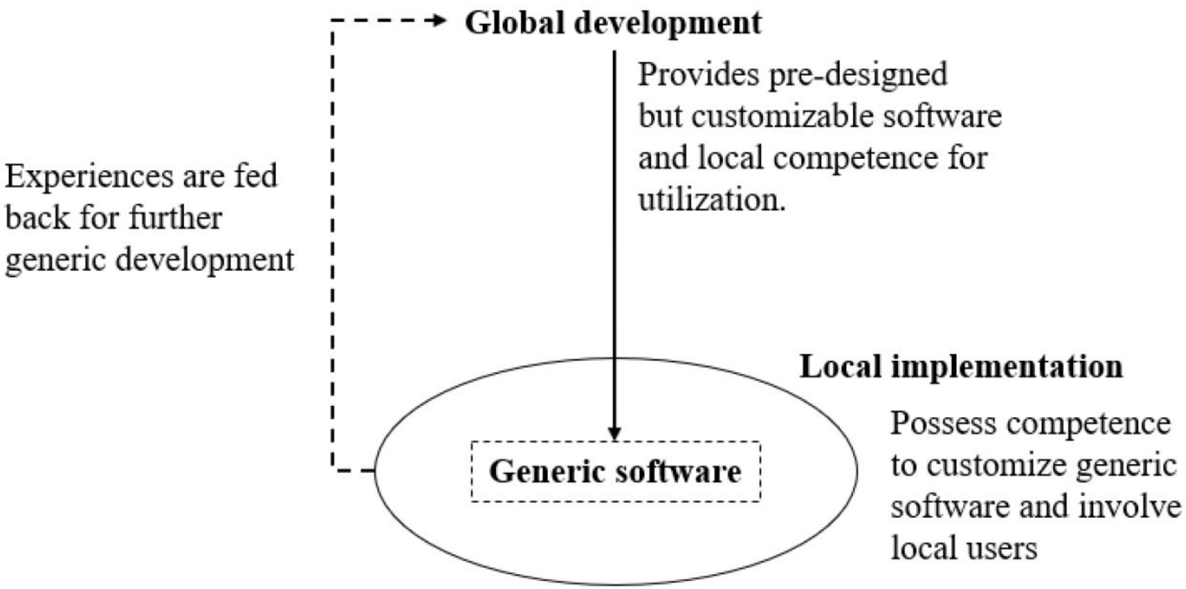


Figure 3-5 Implementers as boundary spanners (Titlestad et al., 2009, p. 18)

Architecture for design



Example: Commodity ordering in Uganda

- Implemented DHIS2 to support commodity consumption reporting and ordering
- Hard to customize DHIS2 to this domain using built-in customization tools
- Platform core → too low use and design flexibility for this case
- Decided to build a third-party app (high design-flexibility)

- Enabled us to create a system tailored to the use-case.
- New tensions on the “scaffolding” and the boundary spanners