Direct Combination

A New Model for Pervasive User Interaction

Simon Holland
s.holland@open.ac.uk
Department Of Computing
The Open University
Milton Keynes

Outline

- *Problem*
- Non-Solutions
- The *Principle* of Direct Combination
- A Sample Domain & *Prototype*
- Theory & Predictions
- A Preliminary, Formative *Evaluation*
- Results of Evaluation
- A Distributed *Infrastructure* - pitfalls?
- Just how flexible? Case Study
- Multiple *Viewpoints* to support user diversity
- Conclusions

Alternative Outline

- *Principle*
- *Generalise & Recurse*
- *Application Channels*
- New Interaction Techniques
- Infrastructure Alternatives

  (simple, conservative vs radical, incremental)
- Viewpoints
- New Ecology

Problem

- Countless, tiny, cheap, wireless-networked devices everywhere (Weiser, 1991).
- Given *ubiquity* and *mobility*, users need to get *two or more devices/resources to inter-operate together* (perhaps unfamiliar, remote).
- Problem: how to make ubiquitous simple powerful, flexible, usable for end-users to control?

Problems 2

- End users often need for rapid interoperation & configuration of two or more unfamiliar devices/resources - find hard (Newman et al., 2002)

- Pervasive environments will often be unplanned, & fast changing as both static & mobile elements change. (Edwards & Grinter, 2001)

- Greater need for end-users to improvise solutions without IT staff. (Kristoffersent & Ljungberg, 2000, Newman et al., 2002)

- Current infrastructures and programming architectures poorly support spontaneous interaction. (Banevar et al., 2000)

Problems 3

- **Resource-poor User Interfaces**
  - Limited screen real estate etc.
  - Limited input device bandwidth/precision/convenience
  - Resource poor interfaces demand time and attention

- **Minimal Attention Situations**
  - User may have limited attention
  - May be walking, driving, shopping, minding children, etc
  - Hands, eyes, ears, may be busy elsewhere (Pascoe, Ryan & Morse, 2000)

Non-Solutions

- Standardisation of protocols?
  - Continual technical changes in devices & standards
  - Competing, overlapping, older vs newer, standards & manufacturers
  - New people & their devices moving through environments
  - Non centrally-coordinated changes in shared environments
  - Unforseen interactions improvised (Newman et al., 2002; Banevar et al., 2000)
- Universal remote standard? (e.g. V2, Zimmerman et al., 2002)
  - Combinatorial explosion
- Context-aware approaches - hard, 'AI-complete' problem (Erikson, 2002)

Problem Summary

- Given *mobility* within *ubiquitous*, *diverse*, *distributed* environments
- *Combinatorial explosion* of capability & possible actions
- Interoperation of devices yields immense search space
- Interoperation with as yet undesigned devices with novel protocols
- Irrespective of any possible developments on standards etc,
  serious UI problem
- No established UI approach is adequate

- New UI principles needed

Possible Solution

Principle of Direct Combination (1999, 2002, 2004)

Scenario: Harry Potter & Direct Combination Wand

Harry raised his wand towards the menacingly advancing Gator and tried to remember the spell for turning it into harmless pig or even a spider. It was no good, he just couldn't remember the right spell ….

Scenario continued

Harry suddenly remembered that this was one of the new Direct Combination Wands. He wouldn't need to recall the spell. Quickly looking around, Harry noticed a small stone on the floor. Pointing the wand at the Gator, Harry made the select *gesture and then made a second* select *gesture at the stone. A glowing list next to the wand presented the two available actions applicable to this particular* pair *of things:* propel the stone at the Gator *and turn* the Gator into stone. *Gratefully Harry activated the second command and the Gator froze into grey immobility.*

DC gives family *of interaction patterns*

1/ Glowing letters list most common general spells, and categories and help facility.

2/ Point wand at Gator with select *gesture, list of most common spells applicable only to Gators.*

3/ Second select gesture at stone. List shrinks to available actions applicable to this particular pair of things.

Insight - Direct Combination Principle

User *always permitted (but not required)* to indicate *two or more interaction objects* involved in intended interaction before being obliged to choose an action.

System can use this to constrain search space of possible commands.
System presents space of focused relevant options to choose from.

Direct Combination always gives user the freedom to specify the parts of commands in *any* order desired, e.g. *noun noun*, *noun verb*, ect.

Allows rapid browsing of object combinations (+- verbs), with actions relevant to choices offered instantly.

Direct Combination Principle (semi-formal)

- UI must always *permit* the user to select ***zero, one, two, three or more*** objects (before any obligation to choose action).
- UI must instantly display actions applying to *particular collection* of objects.
- User chooses an action
- Rapid, Google-like browsing of object combinations

NB - DC interaction styles do not obstruct traditional interaction styles - subsume them

Expressivity of DC

Generalise notion of 'device/resource'

Virtual & Physical, Whole & Subparts,
Visible & Invisible (resource discovery),
Near & remote.

Generalise notion of selecting resource

Physical pointing, Tangible selection,
Remote pointing, Naming, Speech.

Preview - Supporting Framework Issues

- Scaleability, realistic, practical to provide?
- Cheap, simple to maintain (analysts, designers, programmers providers)?
- Flexible under rapid change?
- In whose interest to build/maintain?
- Who decides what interactions are available?
- What if different users want different interactions?

Broad rationale for DC

- Cheap fast way to browse & constrain search space
- Increases, not restricts user freedom
- Distribute user interface around environment
- Recognition vs recall
- Systematic source of simple affordances

Tame search spaces with combinatorial *implosions*

DEMO

scenarios

Scenarios

1. Pipe sound from the Office Radio to myPDA.
2. Display time from the Room Clock onto aWallscreen.
3. Pipe sound from myPDA onto the Car Radio.
4. Get Teddy to vocalize a TextFile over the Car Radio.
5. Arrange it so that if the Front Door opens, the Room Light flickers.
6. Arrange it so that when SJH's Car gets closer than 15 miles to Home, the Room Heating is turned on.

Broad rationale for DC

- Cheap fast way to browse & constrain search space
- Increases, not restricts user freedom
- Distribute user interface around environment
- Recognition vs recall
- Systematic source of simple affordances

Tame search spaces with combinatorial *implosions*

Testing DC Principle

Principles not always easy to test, but…

For *particular systems*, particular interaction styles, domains, users, tasks, situations:

- Usability,
- Capacity to reduce: search space, attention, mental load.

First empirical study (preliminary, formative).

Evaluation

- Formative, preliminary study
- Hypotheses, Analytical Arguments
- Prototype & Domain
- Tasks
- Assumptions about Subjects
- Subjects
- Training
- Protocol
- Data Collection
- Results
- Interpretation

Hypotheses about DC

- Reduce the amount of search required by the user.
  other things being equal.

- Faster, less frustrating, and impose less mental load on the user

- Quick to learn

Or…

Greater variety of interaction patterns will increase load,
slowing user down?

Analytical Arguments

- Situations with many commands, DC often reduces search space.
- DC typically reduces number of interface actions required.
- N >= 3 typically large search reduction.
- Tasks requiring composed actions - typically large search reduction.
- Tasks involving intermediary tools.

Formal modelling vs empirical testing

Domain & Prototype

- Simulated Ubiquitous environment modelled on laptop
- Wide range of selectable objects in environment
- Appropriate behaviour & state coded for each type of object

- Simulated DC wand on laptop
- Outcomes displayed in dialog boxes & states inspectable

- Interface can work in DC or non-DC mode.
  (Non DC mode is effectively standard everyday UI)

- *Recall - DC can apply to any combination of physical objects, virtual objects, remote objects, subparts of objects*

Subjects

- 8 subjects (4 pairs of 2, to promote think-aloud)
- Recruited by availability: & ensuring variety of education level, computing experience, age
- None had seen or used a DC interface

Tasks

1. Pipe sound from the Office Radio to myPDA.
2. Display time from the Room Clock onto aWallscreen.
3. Pipe sound from myPDA onto the Car Radio.
4. Get Teddy to vocalize a TextFile over the Car Radio.
5. Arrange it so that if the Front Door opens, the Room Light flickers.
6. Arrange it so that when SJH's Car gets closer than 15 miles to Home, the Room Heating is turned on.

Assumptions about Subjects

All subjects should:

- be familiar with *conventional user interfaces*.
- be familiar with Ubicomp idea: *every object is networked*; PDA or wand can *select* and *control* objects, locally and remotely.

- have exposure to basic *idea or principle of DC* as expressed in a few sentences.
- have heard one or two *short stories* or scenarios to make idea of DC memorable.

Last three assumptions met by elementary *Training* ->

Protocol

- Sequence of 6 tasks, carried out in pairs
- Think aloud
- 6 tasks per condition (DC and non-DC).
- Same tasks each condition, but balanced design
  (half subjects DC first, half non-DC first)
  (to allow for any learning & interference effects)

Training

Three single sheets to read, to meet assumptions

No demonstration of operation given

Scripted tour - slight difference DC vs non-DC conditions

Scripted oral screen tours showed:

- what pane to use to select simulated objects,
- what pane to use to see the objects selected,
- what pane to use to select relevant commands.

Data Collection

- Observation, video taping of screen, sound recording.
- Task completion times.
- Subjects each singly use NASA TLX human workload index to measure 5 dimensions after each task, (mental demand, effort, frustration, temporal load, physical load).
- Short questionnaire each subject at end of session.

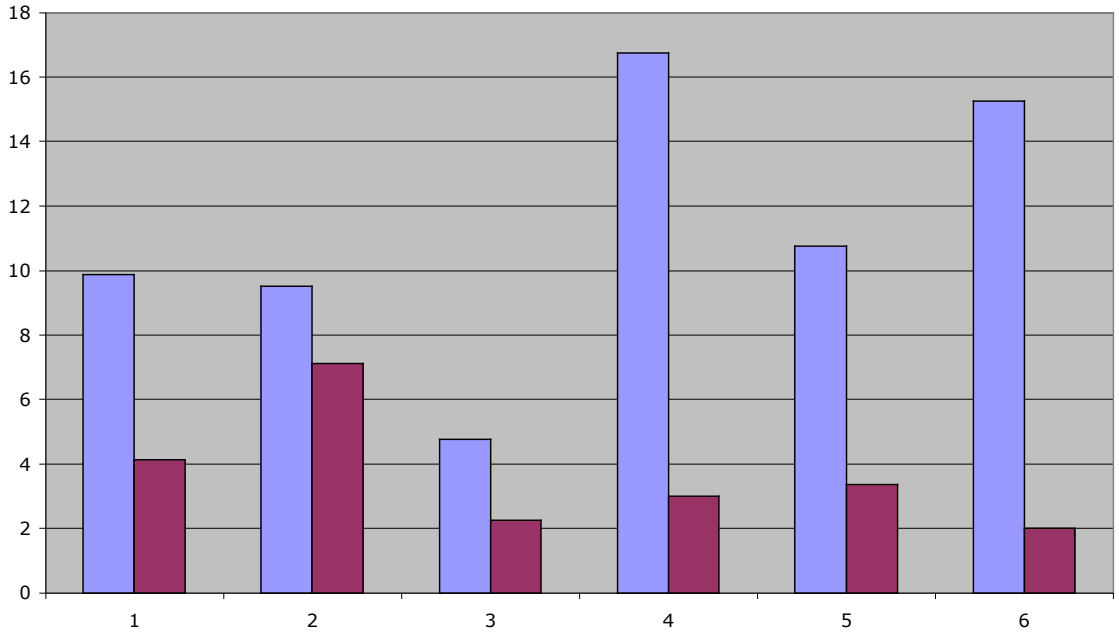
Results 1

NASA TLX Human workload index

Averaged over all subjects,
for all 6 tasks,
for all 5 workload measures
(Mental Load, Effort, Frustration, Physical Load, & Temporal Load)

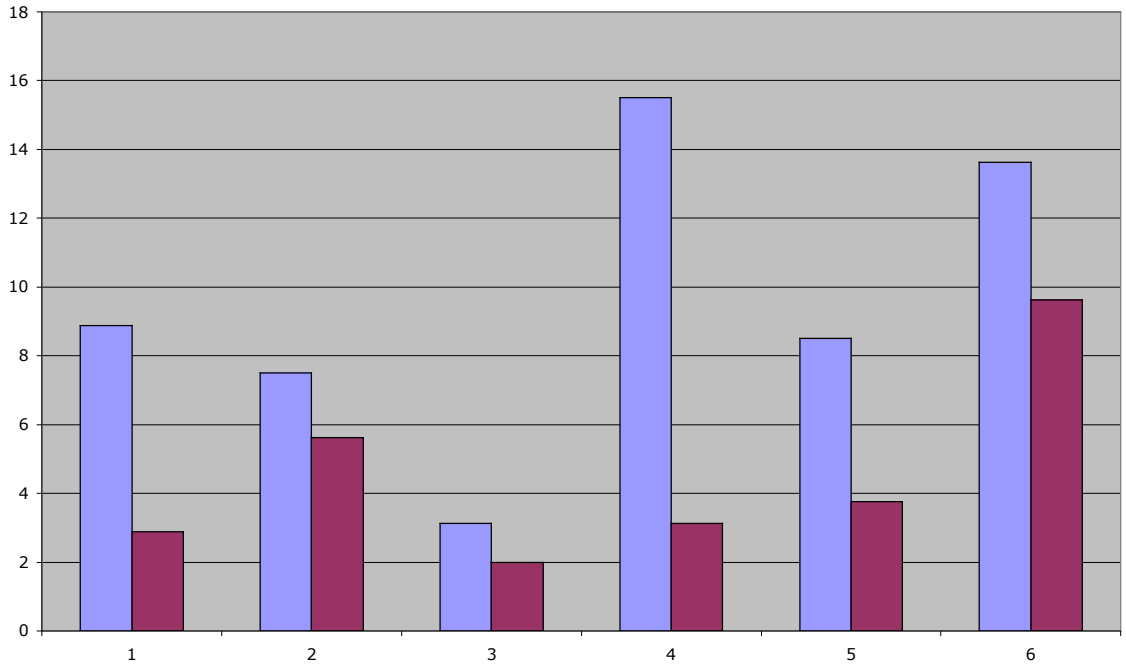
Loads for DC condition lower *than for reference condition*.

Mental Load



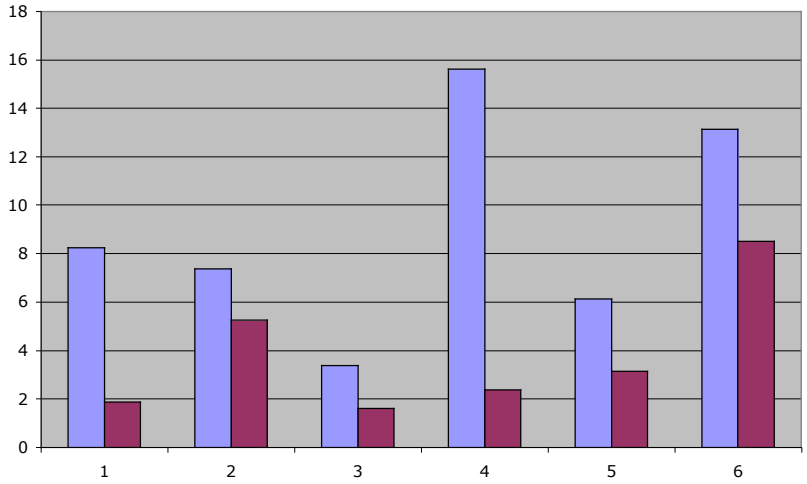
Mental Load for DC vs non-DC, on a 20 point scale,
as measured by the NASA TLX (averaged over all subjects)
for tasks 1-6. DC scores are shown in the darker shade

Effort



Effort for DC vs Non-DC, on 20 point scale, as measured by NASA TLX.
Average ratings over all subjects for tasks 1-6. DC scores are shown in darker shade.

Frustration



Frustration for DC vs non-DC as measured on a 20 point scale by NASA TLX (average ratings for all subjects) for tasks 1-6. DC scores are shown in the darker shade

Results 2

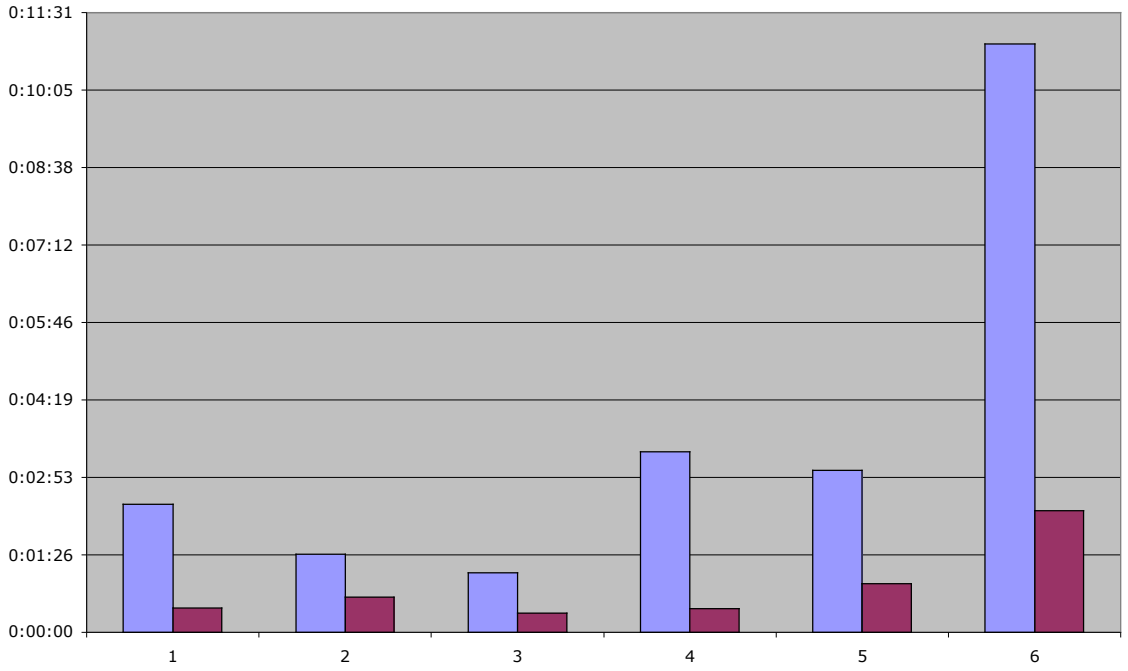
Task Completion Times

Averaged over all subjects,
for all 6 tasks,
time taken to complete each task
(or until task abandoned) was

Less (faster) *for DC condition than for reference condition*.

Given think aloud, timing must be treated with caution.

Task Completion Times



Time in mins and secs to complete tasks for DC vs non-DC.
Averages for all subjects shown for tasks 1-6. (Max time about 11 mins). DC in darker shade

Results 3

3. Questionnaire

On all questions asked,
all modal (most popular) ratings judged DC condition more favorably than reference.

Direct Combination

Direct Combination applied to user interfaces tends to:	Disagree Strongly	Disagree	Disagree Weakly	Neutral	Agree Weakly	Agree	Agree Strongly
reduce the degree of search required to carry out tasks		1		2	1	4	
reduce the amount of time required to carry out tasks		1			1	4	2
reduce the amount of attention required to carry out tasks				2	1	4	
reduce the amount of work required to carry out tasks				2		4	1
reduce the amount of frustration in carrying out tasks				1	2	2	2
lessen the need for memorisation in carrying out tasks				1	2	4	1
lessen the demands of interface navigation			1	1	1	4	1
reduce the amount of stress involved in carrying out tasks			1	1	1	4	1

Interpretation

- Formative, preliminary small scale, first evaluation.
- Preliminary indications strong, consistent & in same direction.
- Users had used conventional interfaces many times, but never DC.
- Preliminary consistent support for analytical predictions.

Limitations of Evaluation

Formative
Not enough users for reliable stats
Think aloud confounds reliable timing
Not rigorous, but useful preliminary indication

Conclusions from evaluation

- Analytical arguments & formal models identify several distinct ways in which DC can reduce search *(some clarified by this study)*.

- Formative evaluation - preliminary, but consistent evidence:
  - DC Usable for users from a variety of backgrounds,
  - DC Rapid for diverse users to learn,
  - DC Faster, less effort, less frustrating, less mental load,
  - No evidence of unexpected penalties.

Moving to DC in the Large Distributed Infrastructure for DC

- How to support large scale distributed use
- Identifying and addressing any potential problems with infrastructure

Supporting Framework Issues - revisited

- Scaleability, realistic, practical to provide?
- Cheap, simple to maintain (analysts, designers, programmers providers)?
- Flexible under rapid change?
- In whose interest to build/maintain?
- Who decides what interactions are available?
- What if different users want different interactions?

Requirements for DC infrastructure

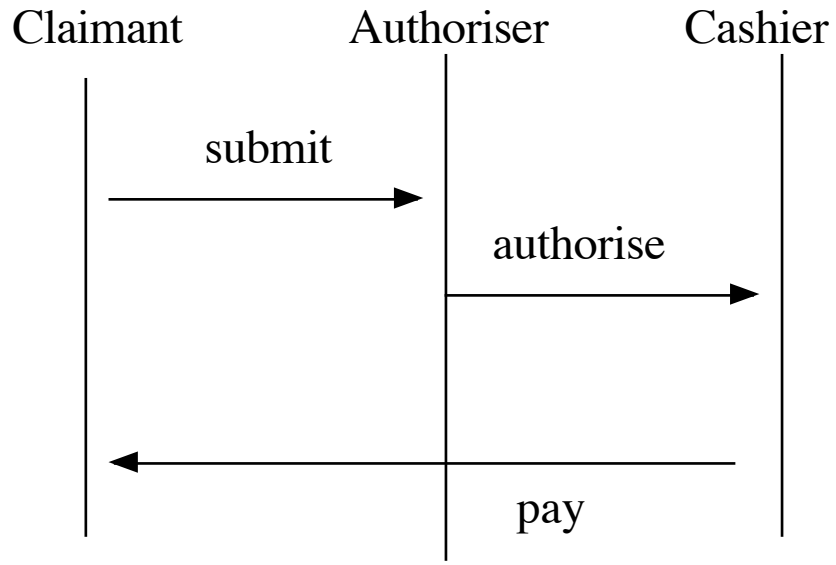
- Simple, practical for analysts, designers, programmers providers.
- Flexible under rapid change.
- Cheap and easy for anyone to publish, refine, amend suites of interactions.
- Easy for diverse end-users to be offered different interactions (even on the fly) to fit current role.

Outline of proposed *distributed* infrastructure

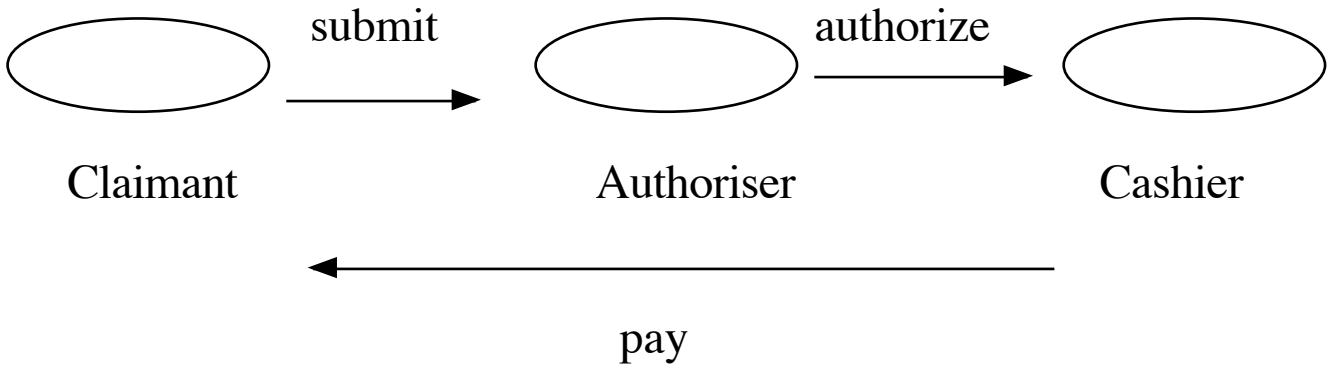
- 'Class'- unique ID assigned to a type of object by manufacturer (or stake holder managing environment in which objects used).
- Role-based framework - allow large range of interactions to be managed in flexible, open-ended, incremental way.

Role-based Analysis & Design

Dewey (1929), OORAM, Reenskaug (2001)

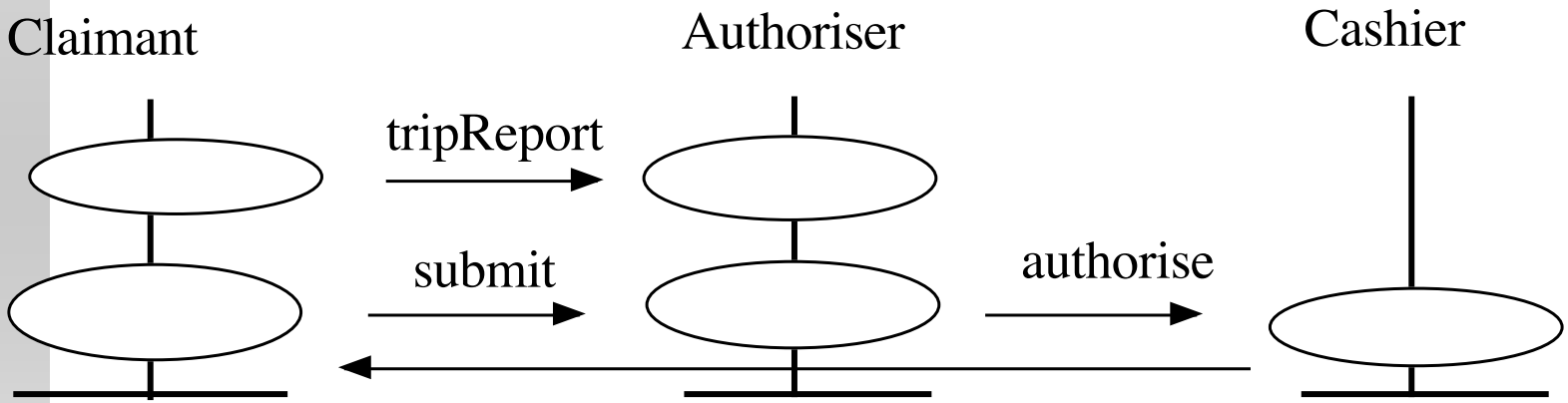


Role-Model



Reenskaug (2001)

Class assembly via hat-stand model

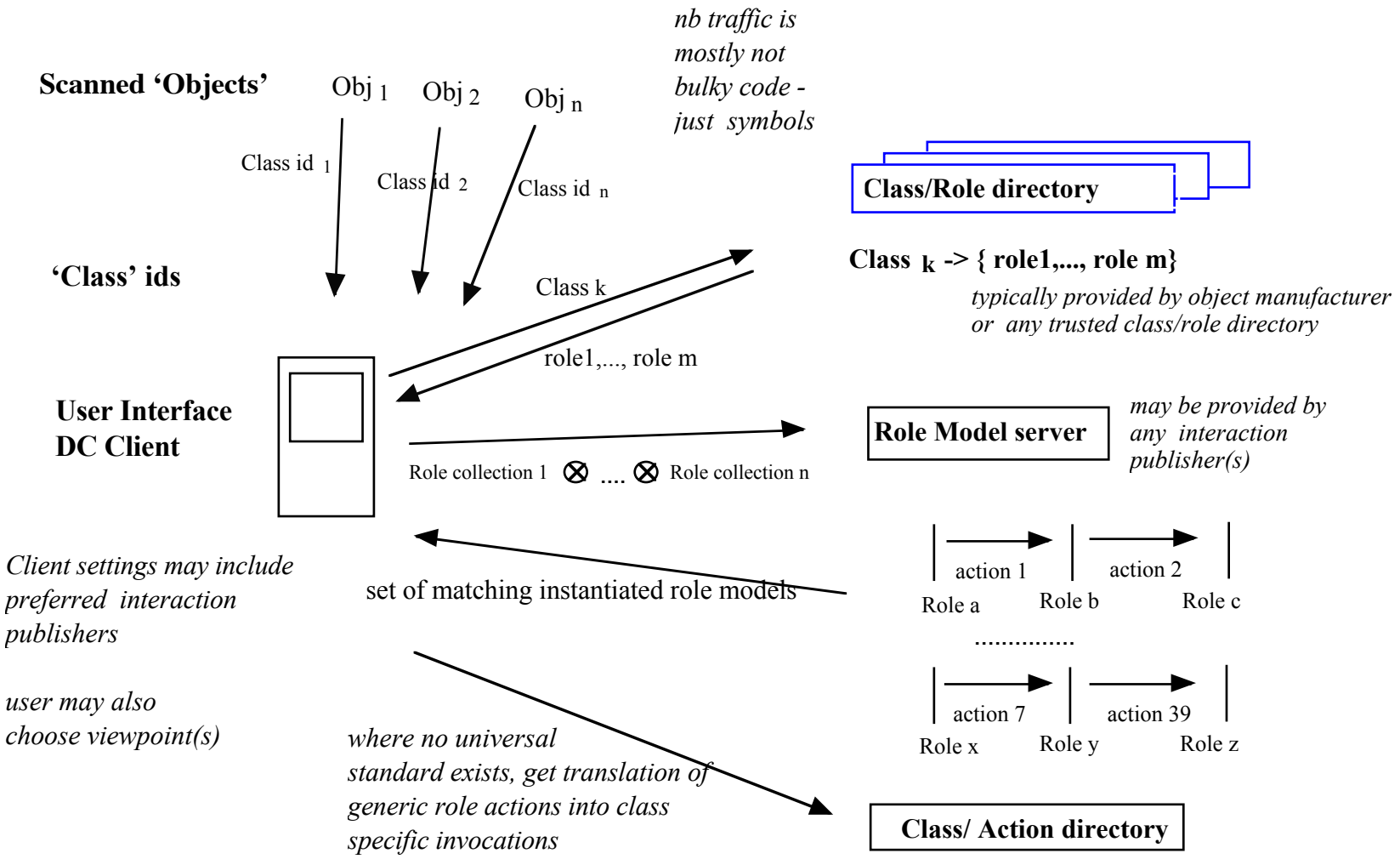


DellaFerra, P, in Reenskaug (2001)

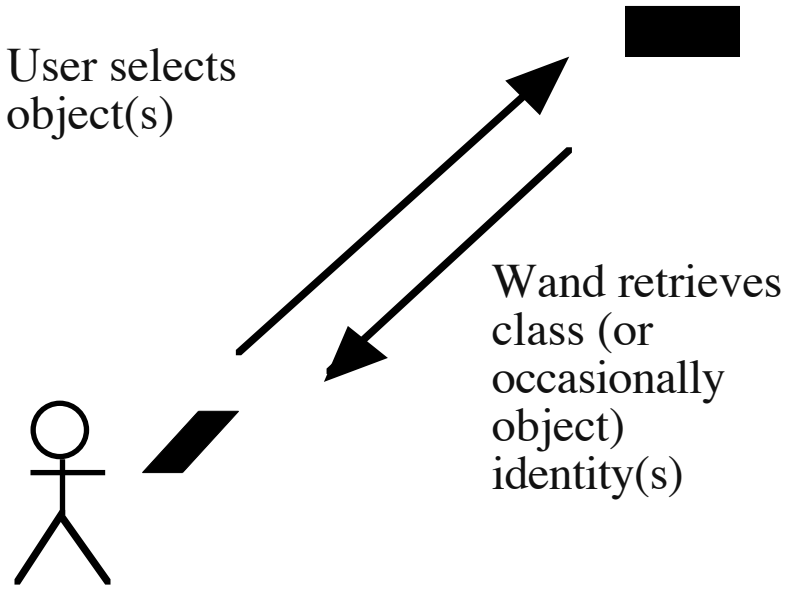
About roles

- For each class of device,
- Identify roles & role models (thin slices of behaviour linking roles, abstracted from classes
- A role model can do just enough to support one meaningful interaction
- Roles are narrow & so highly reuseable.
- But roles are abstract & so widely applicable to many classes
- 'Hat stand' model of classes -assemble classes by collecting roles.
- Role names must be unique (at least in a given naming space)

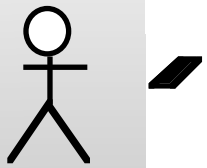
Direct Combination

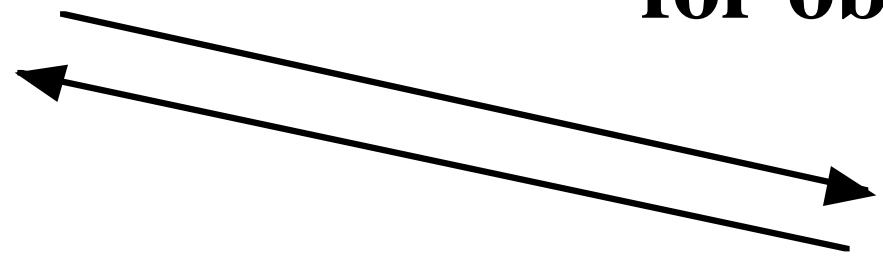


Role- based DC Infrastructure step by step

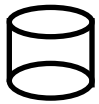


Get currently associated role names for object





Role server



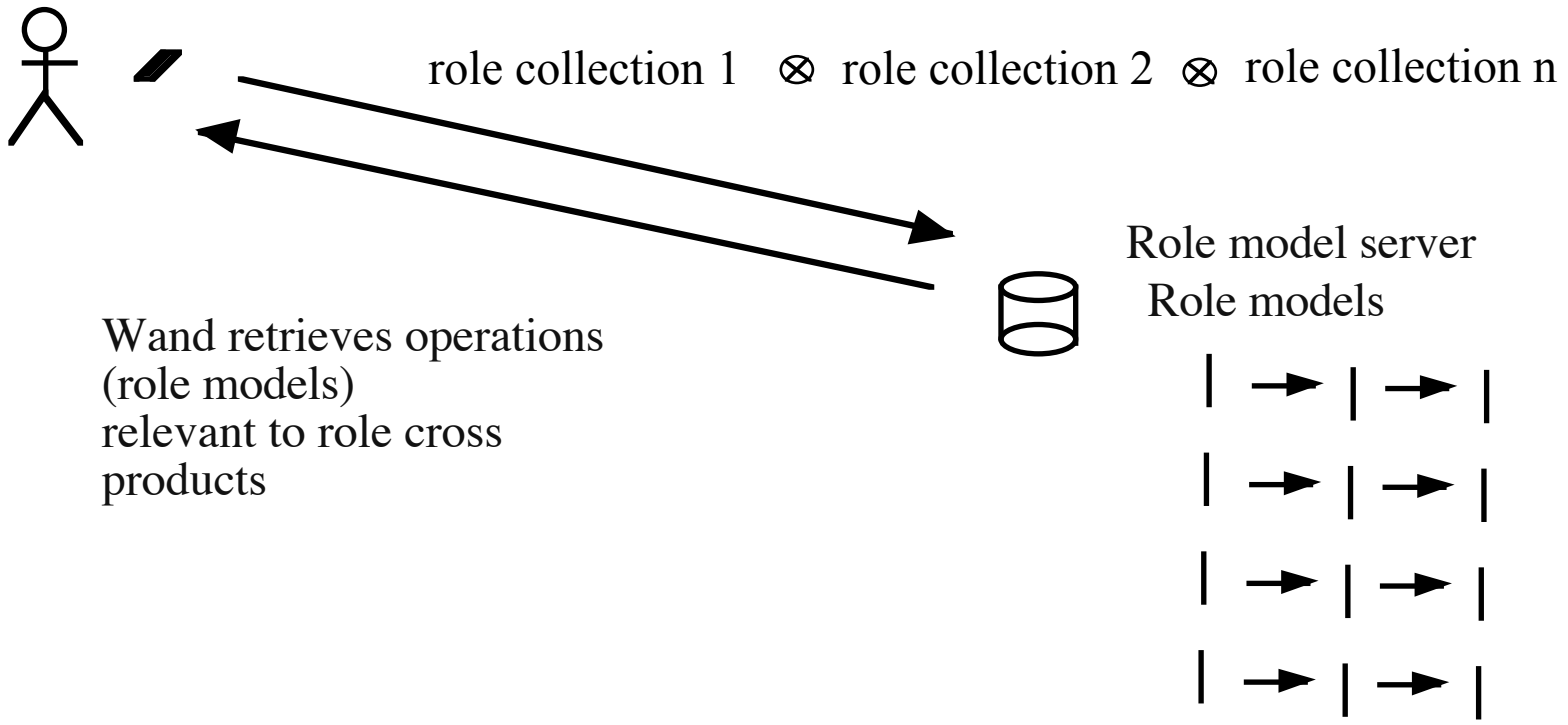
Class i
role1
role2
role3

Wand retrieves
roleNames
for class or object

Typically provided by object manufacturer or service provider or other trusted party

Additional roles may be added by other trusted parties

Use role models to find actions relevant to *pair* of selected objects



User chooses operation from choices

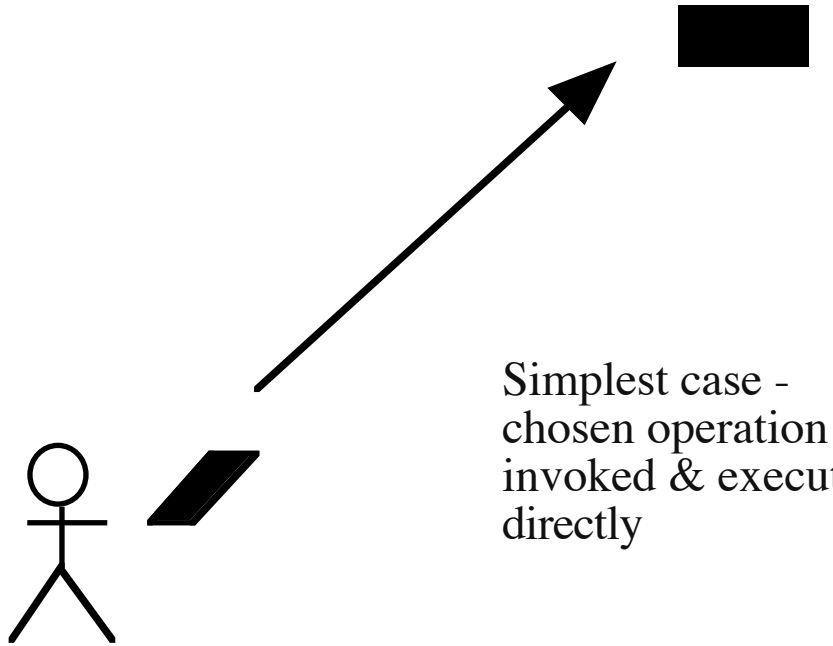
Do X with Object1 and Object2

Do Y with Object1 and Object2

Do Z with Object1 and Object2

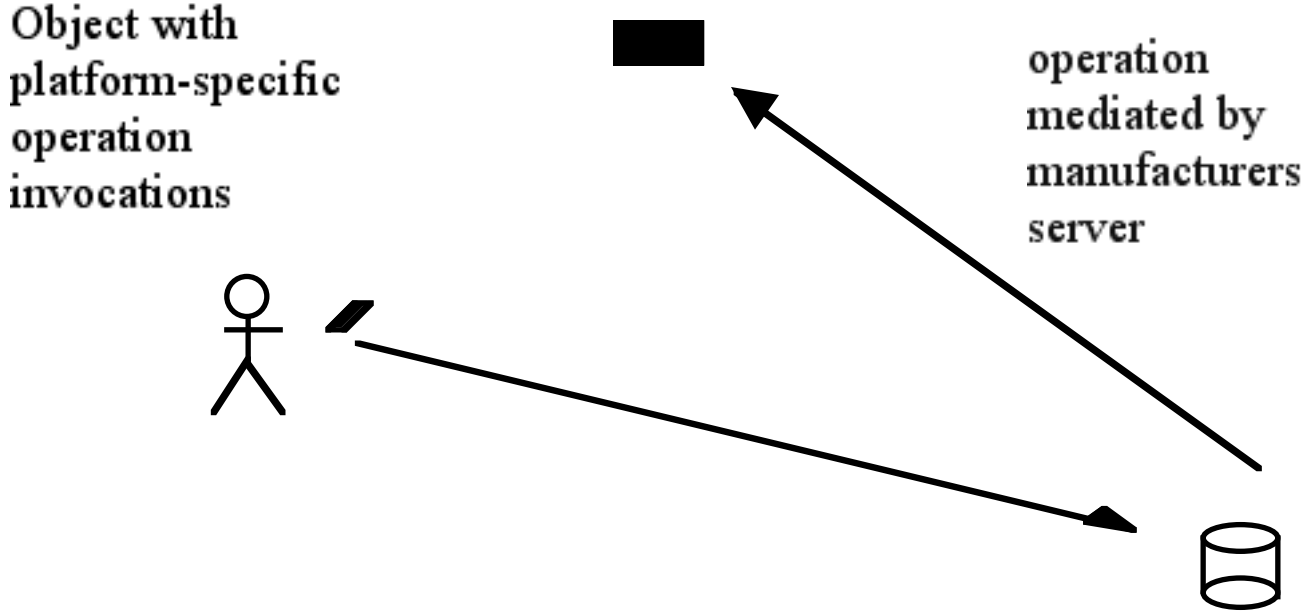
- For simple actions -that's all.
- In more complex cases, role model may download simple helper program or generally behave loosely like an applet

Role- based Architecture



Simplest case - chosen operation invoked & executed directly

Platform-specific operation invocation



Implications of infrastructure

• Useful new roles based on existing class capabilities may be added incrementally to support new interactions with other devices as new role models are identified.

• Objects need be able to play promised roles, hence users should only use class/ role directories (or extensions thereof) provided by trusted responsible parties such as:

  • Manufacturer of the class in question,
  • DC service provider,
  • or any other interested trusted party.

• *Some* objects may have the capabilities to play a role, but may not be able to execute commands given only the name of a *role action*.

•Trusted interested parties such as the manufacturer can provide servers 'Class/Action servers' to translate role names where needed.

Hypothetical case study

Burp-vocalizing teddy

Burp vocalizing teddy

10 am Step 1: Behaviour fragment identified

11 am Step 2: Remote invocation method identified

12 am Step 3: Role name & action registered

burb-vocalizer burp-vocalize

2 pm Step 4: Role model published on student union role model server

vocalizable burb-vocalizer burp-vocalize

4 pm Step 5 Viz, Slashdot etc servers add

- *burb-vocalizer* to speaking-teddy class definition
- *burp vocalize* role model
- translation of role action to remote invocation

added to class definition for teddy

NB implications for interactions available to ALL *vocalisables*

Next Day Routes by which teddy may acquire burp-vocaliser role

Routes by which teddy may acquire burp-vocaliser behaviour

UK

2pm Manuf adds newly defined role to UK class def for teddy

2.15 pm Some parents organisations block this role and role model via house proxy link to *role-nanny* servers

2.20 Kids whose house servers allow roles to be pasted onto individ objects straight from web do this anyway

Note: Does not work with teddies without the hardware

2.30 It does now, provided a hi fi or radio etc in range using a software emulation.

France

2.10 Manuf declines to add role to French class def for teddy

2:15 Various special interest groups, magazines, publish role and role model for providers that blend in their services

Infrastructure proposal summmary

- Relatively simple, flexible framework
  (analysts, designers, programmers providers)
- Flexible under rapid change
- Infrastructure need not be perfect to be useful
  historical lesson from the web

- Who would create & select roles & role models for objects?

  Organisations, Individuals,
  Magazines, Special interest groups,
  Professions, Manufacturers,
  Children, etc...

Problem: diversity of viewpoints

How can diverse end-users tune a distributed DC system to provide (possibly mutually contradictory) interactions appropriate to their current role?

Viewpoints

A viewpoint is:

A set of mappings from classes to roles, and
A set of role models

(Cruder version - a set global of roles restricting possible role models)

More formally, a viewpoint is:

A set of mappings from classes to roles

$f_i: C_i \rightarrow \{ r_{i1}\ r_{i2}\ ..r_{in} \}$

A set of role models

$\{((r_a , r_b), \{ op_{a1}, op_{a2},..op_{an}\})$

.....

$(r_x, r_y), \{ op_{x1}, op_{x2},\ldots\ op_{xn}\})$

What do viewpoints offer?

Different users may see different (perhaps contradictory) functionality offered by the same object,
And different behaviours.

Same mechanism may be used to deal with security, sharing etc

Viewpoints in practice

Who defines viewpoints?

Manufacturers, special interest groups
Professional groups, News groups
Magazine, Service providers...

Users may choose, combine viewpoints to suit current role

Level of granularity available is simplest behaviour provided associated with some role model

Dedicated wands, subscriptions, passwords may be required to gain access to certain viewpoints

Related work

Several approaches may be seen as limited special cases of DC

- Pick and Drop offers limited special case of DC.
  (2 ops only - get & put) (Rekimoto 1997, 1998)
- InfoStick (Kohtake, Rekimoto et al. 1999) provides
  limited special case of DC.
- DataTiles (Rekimoto, Ullmer et al. 2001)
  Tangible adjacency of tiles affords single operation.
  Applying DC would give far greater flexibility and power.
- DC is generalisation/specialisation of DM (Shneiderman, 1983)

Supporting Framework Summary

Principle (1999, 2002) vs Supporting Framework (1999 - 2004)

Supporting Framework

New Interaction Styles
Software Architecture
Personalisation, Roles, Viewpoints
Domain Analysis & modelling framework
Particular applications

Well-founded framework mechanisms (2003, 2004)

- role-based architecture • multiple viewpoint mechanism,
- sound n-dim DC algorithm, • domain analysis semantics

Related work 2

Similar *goals* to DC, but entirely different approaches

- Proem Project
- Recombinant Computing (Edwards, Newman et al. 2002)
  Speakeasy Project
- Aura architecture (Sousa and Garlan 2002)
- Frameworks for Tangible computing (Ulmer & Ishi, 2000)
- iRoom tuple-space approach (Borchers, Ringel et al. 2002)

Conclusions

- DC is a new UI principle.
- Analytical arguments & formal models identify several distinct ways in which DC can reduce search *(some clarified by this study)*.
- DC especially useful when end user must manage *two or more resources* (digital or physical) *to interoperate*.
- Prototype implementation of scaleable, multiple-viewpoint framework.

- Formative evaluation - preliminary, but consistent evidence:
- DC Usable for users from a variety of backgrounds,
- DC Rapid for diverse users to learn,
- DC Faster, less effort, less frustrating, less mental load,
- No evidence of unexpected penalties.
- Principle of Direct Combination widely applicable.

END