Building Product Populations with Software Components

Rob van Ommering
Philips Research
May 23\textsuperscript{rd}, 2002
Product Populations

Software Components

Building ,, with ,, 
A Television Product Family

Price

Region

Output Device

Other axes of diversity: image, sound, data processing, user interface, connectivity, …
Other Product Families

- VCR
- STB
- DVD
- Audio
Convergence

TV + VCR = TVCR

TV + DVD = TV-DVD

TV + HD = Tivo

TV + STB = Digital TV

TV + Audio = Home Theater
A product population is:
- a set of products with many commonalities,
- but also with many differences,
- developed by different suborganizations,
- each with its own time-line / lifecycle.
Product Populations

Software Components

Building ,, with ,, 
The Koala Component Model

Koala is:
- a component Model
- with an ADL
- to build populations of
- resource constrained products
A Product Line for Bears

...
Provision of Interfaces

tun: ITuner

CTuner

Evolution

tun: ITuner
tun2: ITuner2

CTuner'

Diversity

tun: ITuner
stun: ISearchTuner

CSearch Tuner

Looks like:

Microsoft COM
Requires Interfaces

All context dependencies are made explicit…
…and are bindable by a third party

…so they can be bound differently in another product
'Connectors'

Direct

Switch

Glue Module

Looks like Hardware!

Looks like Visual Basic
The composition process is recursive...

Component instances are encapsulated.
Component types are not (necessarily) (see later).
Diversity interfaces are *outgoing* interfaces which parameterize the component.

Partial evaluation is used to create resource efficient configurations.
Product Populations

Software Components

Building „, with „,
A Real-Life Example
Problem: many (>100) activities but few (<10) threads

Step 1: use message **pumps**
created on virtual pump engines
required through a diversity interface

Step 2: bind these to **pump engines**
(a real dispatcher loop)

Different thread,
Synchronisation required

Same thread,
No synchronisation required
Evolution

Koala subtypes interfaces based on set inclusion of functions

Koala reports an error if a non-existing interface is bound...!

Provide more...

Require less...???
Separate component information from interface information
Separate component external behaviour from component implementation
Use *does* instead of *shall*. 
Distinguish between:
- versions
- temporary variants
- permanent variants

of components.

We use our CM system for:
- versions
- temporary variants

But we use the component model for:
- permanent variants
Concluding Remarks

Product populations arise from ‘convergence’ of products, and are becoming increasingly important.

Product populations require a balance between classical product family development and a COTS-like approach.

Introducing product populations also impacts the software process and organization.

We currently have 100-200 software engineers in 10 sites deploying our approach.

But we still have a long way to go…
Thank you!