Visualising Architectural Dependencies

John Brondum, **Liming Zhu**

National ICT Australia (NICTA)

University of New South Wales (UNSW)

Liming.Zhu@nicta.com.au

Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Funding and Supporting Members and Partners

Motivations

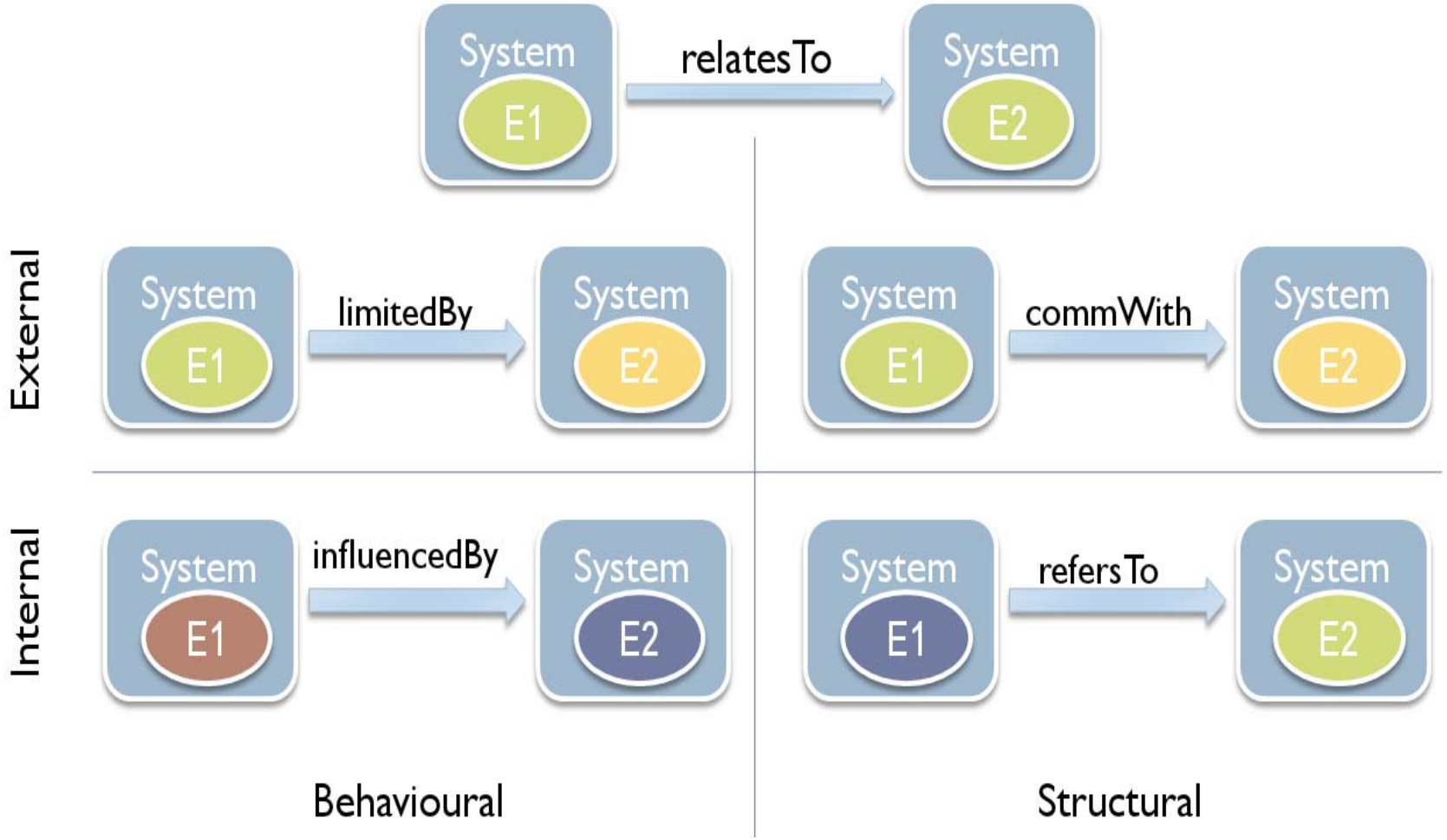
- Architectural dependency has TD potential
- Architectural dependency ≠ software dependency
  - New dependency concepts
    - understandable by all levels of stakeholders
    - not just aggregation of code-level dependency
    - architectural significant
  - "Implicit" dependencies & beyond code
    - Indirect; other factors (context, org structure, knowledge mgt..)
  - Automated analysis of code has limitations
    - large-scale system of systems with black-box components
    - Issues that are undetectable by code analysis
- Visualising them in architecture views is important

Contributions

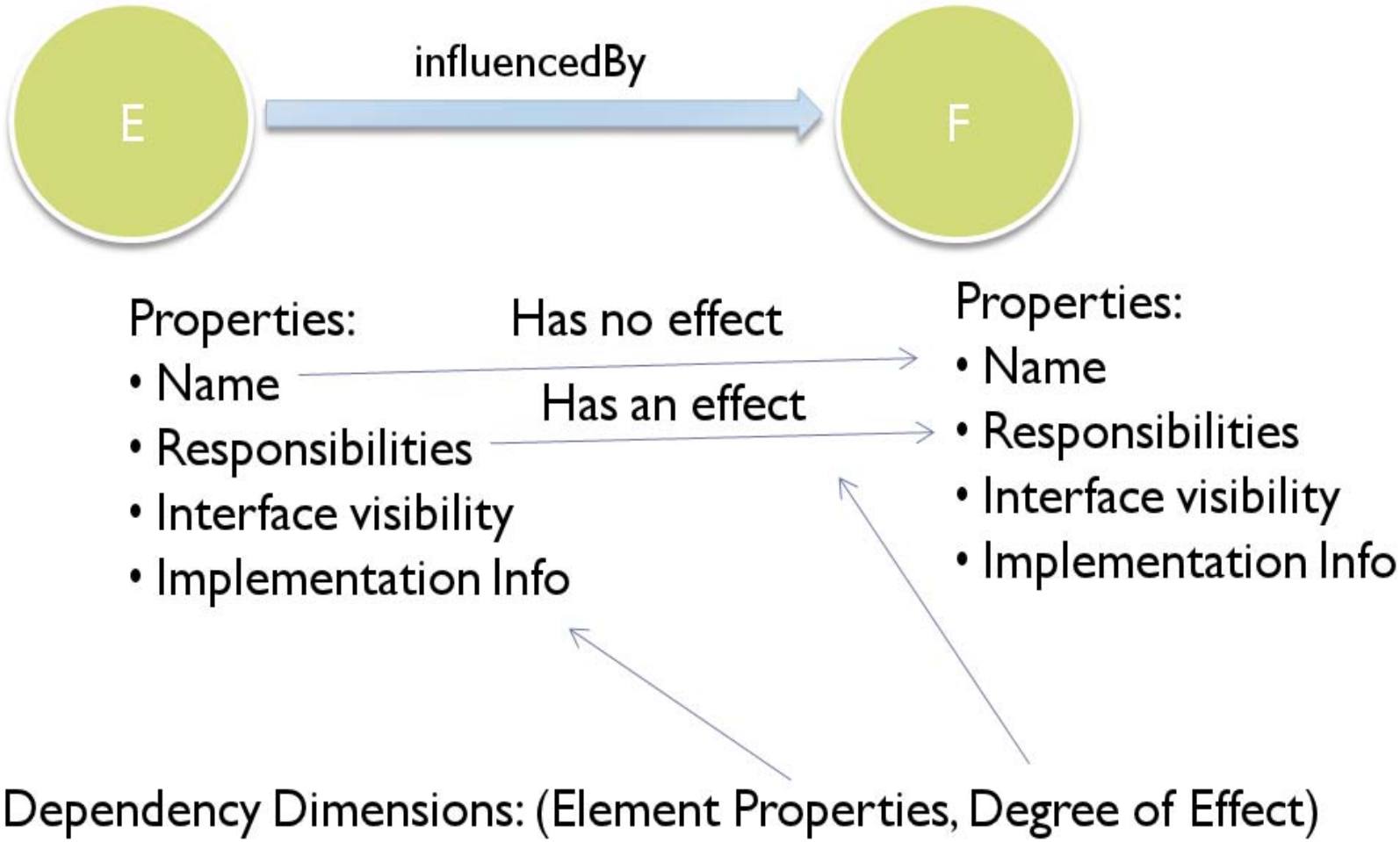
- New dependency relationship types
  - Dimensions & degrees of dependency
  - Cross views & models
  - Relatedness of dependencies: implicit dependency
- Evaluated in three case studies
  - Online Production Systems
    - Implicit dependency → explicit design compromise → explicit debt
  - Learning & Teaching Portal
    - Synchronisation issues → highlighted as new dependency types with dimension→ integration debt
  - Lending Valuation Systems
    - Upgrade problems → omitted dependency → design debt

Note: Analysis method and tool support (submitted to WICSA)

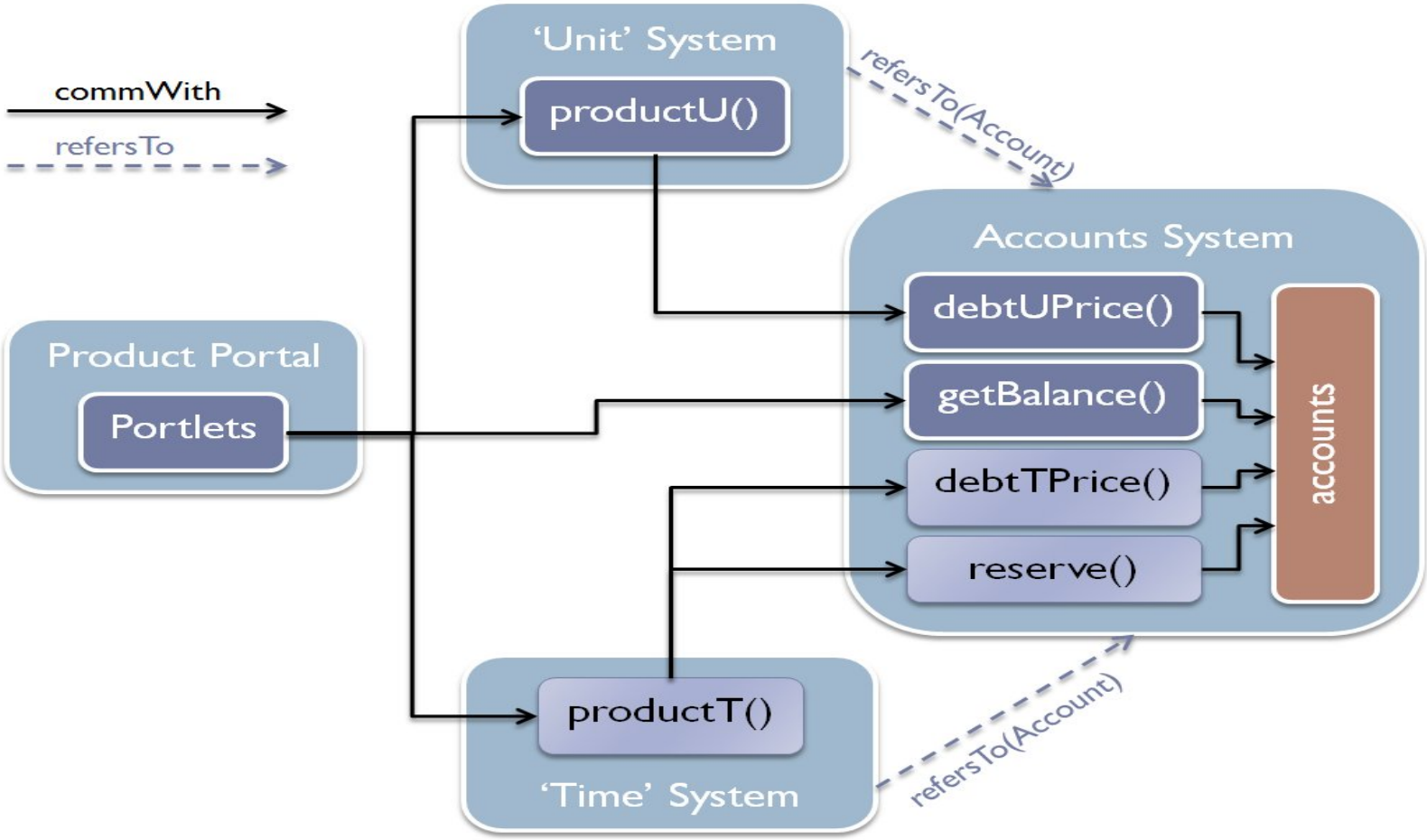
New Dependency Relationship Types



Dimensions & Degrees of Dependency

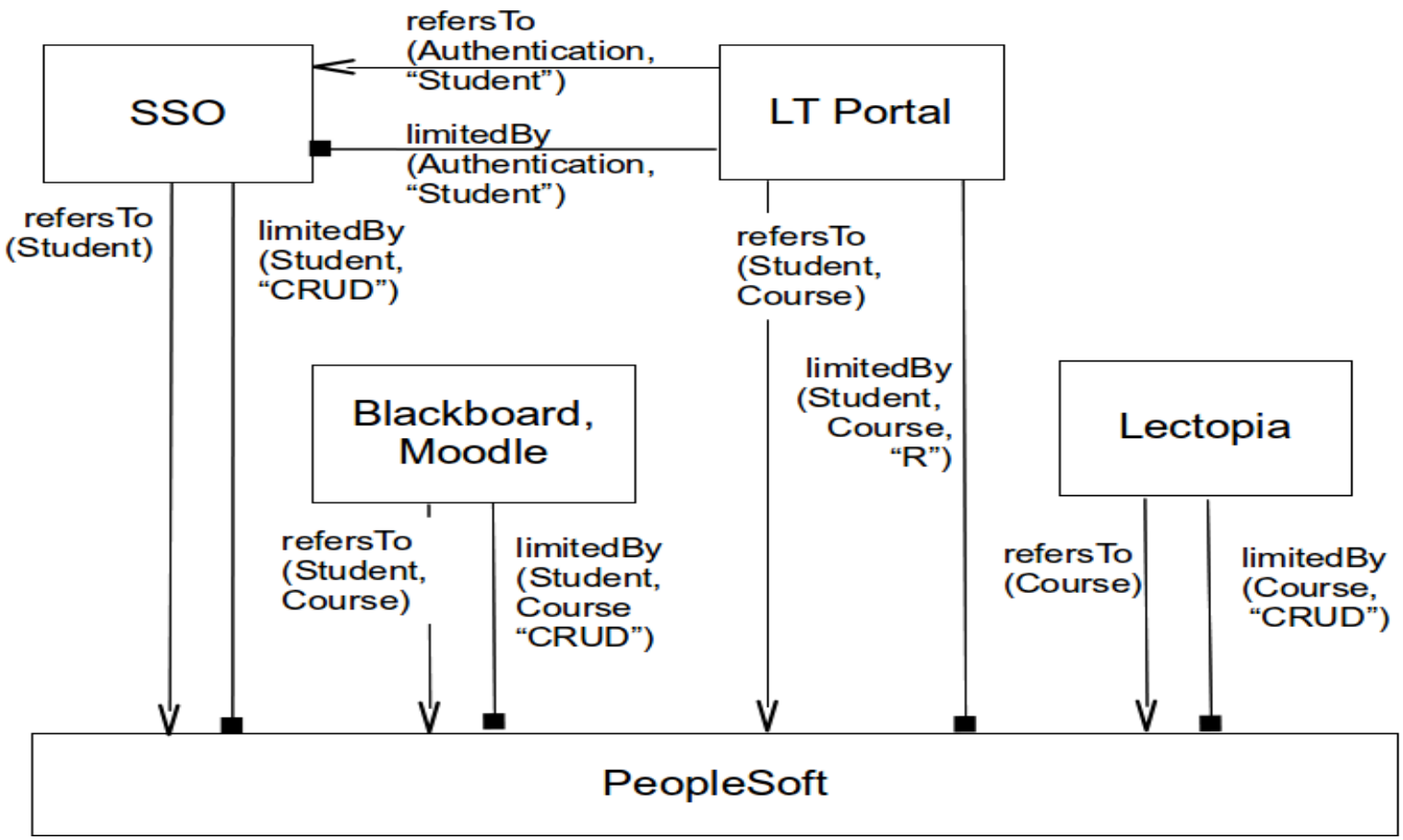


Case 1: Online Product System



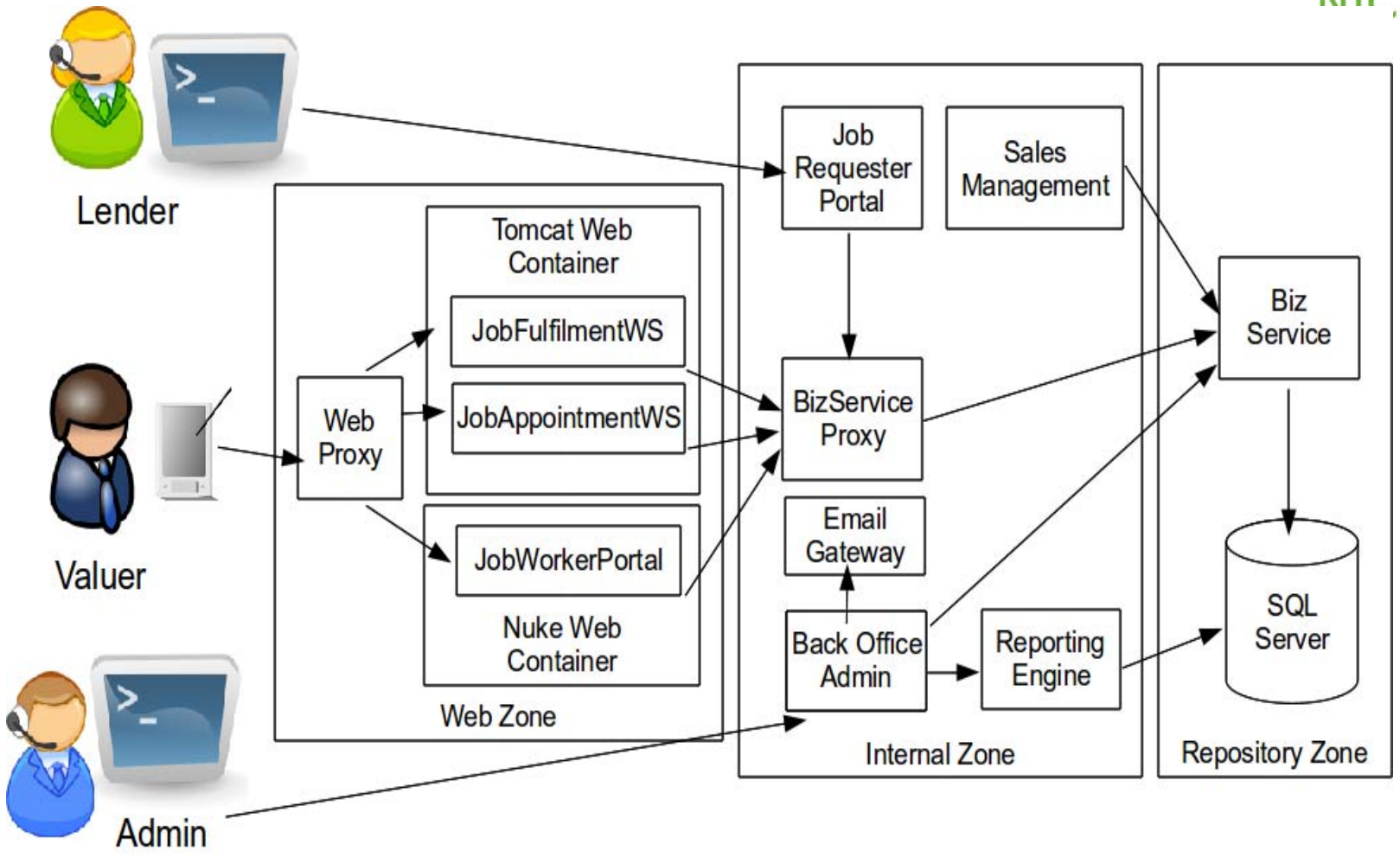
Implicit dependency → explicit design compromise → explicit debt

Case 2: Learning & Teaching Portal



Synchronisation issues → highlighted as new dependency types with dimension → integration mismatch debt

Case 3: Lending Valuation Systems



Upgrade problems → omitted dependency → design debt

DSM Annotated with Dependency Types

	MJA	WP	PW1	eW1	JF	JA	JWP	JRP	BSP	SM	BOA	RE	EG	BS	SSD
Mobile Job App (MJA)	-	C												R	
Web Proxy (WP)		-	C	C	C	C	C								
pdaWeb1 (PW1)			-		C	C									
endUserWeb1 (eW1)				-			C								
JobFulfilmentWS (JF)			C		-	L		I	C					R	
JobAppointment (JA)			C		L	-		I	C					R	
JobWorkerPortal (JWP)				C			-	I	C					R	
JobRequesterPortal (JRP)								-	C	I				R	
BizServiceProxy (BSP)									-					C	
SalesManagement (SM)										-				C,R	
BackOfficeAdmin (BOA)					I						-	C	C	C,R	
ReportingEngine (RE)												-			C,R
EmailGateway (EG)													-		
BizService (BS)														-	C,R
SqlServerDatastore (SSD)															-

Conclusion

- New architecture-level dependency types
  - Top-down motivation
    - not just aggregation of code-level dependency
    - for stakeholders at different technical levels
  - For debt that not easily detectable in code
    - incorporating non-code factors as dimensions
- Evaluated in real world case studies
  - Retrospectively on projects by linking problems with new dependency and design debt
  - Proactively on projects by identifying new dependency and making the debt explicit
- Future work