An Evaluation on Developer's Acceptance of EasySOC: A Development Model for Service-Oriented Computing

C. Mateos M. Crasso A. Zunino M. Campo

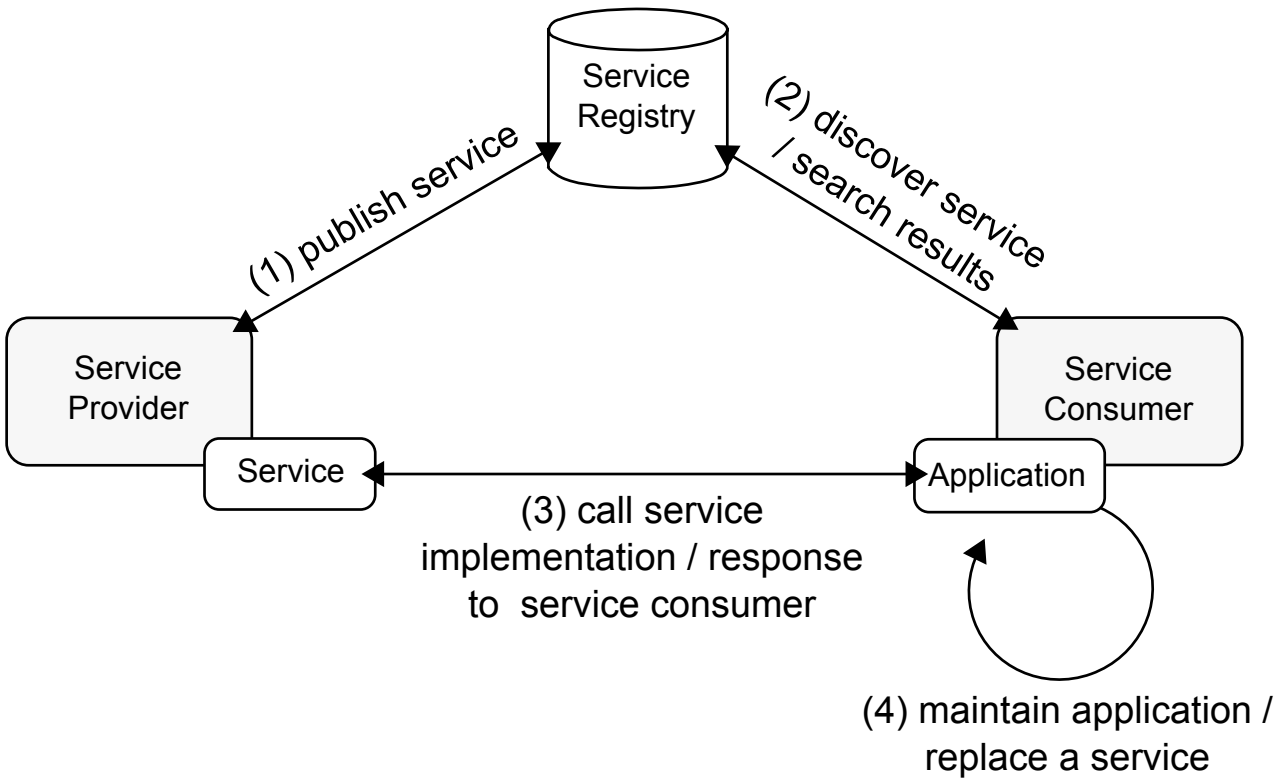
ISISTAN - UNICEN, Tandil, Argentina
CONICET

Argentine Symposium on Software Engineering

Outline

Paradigm Actors & Activities

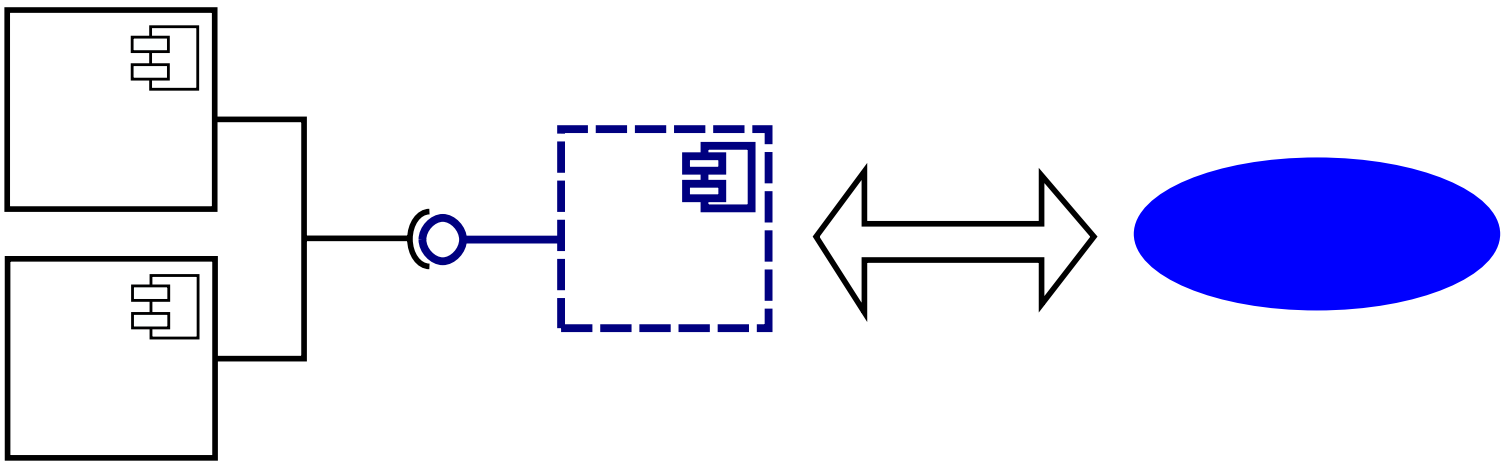
- SOC is a relative new and exciting paradigm for distributed computing



Traditional Steps for Developing Applications

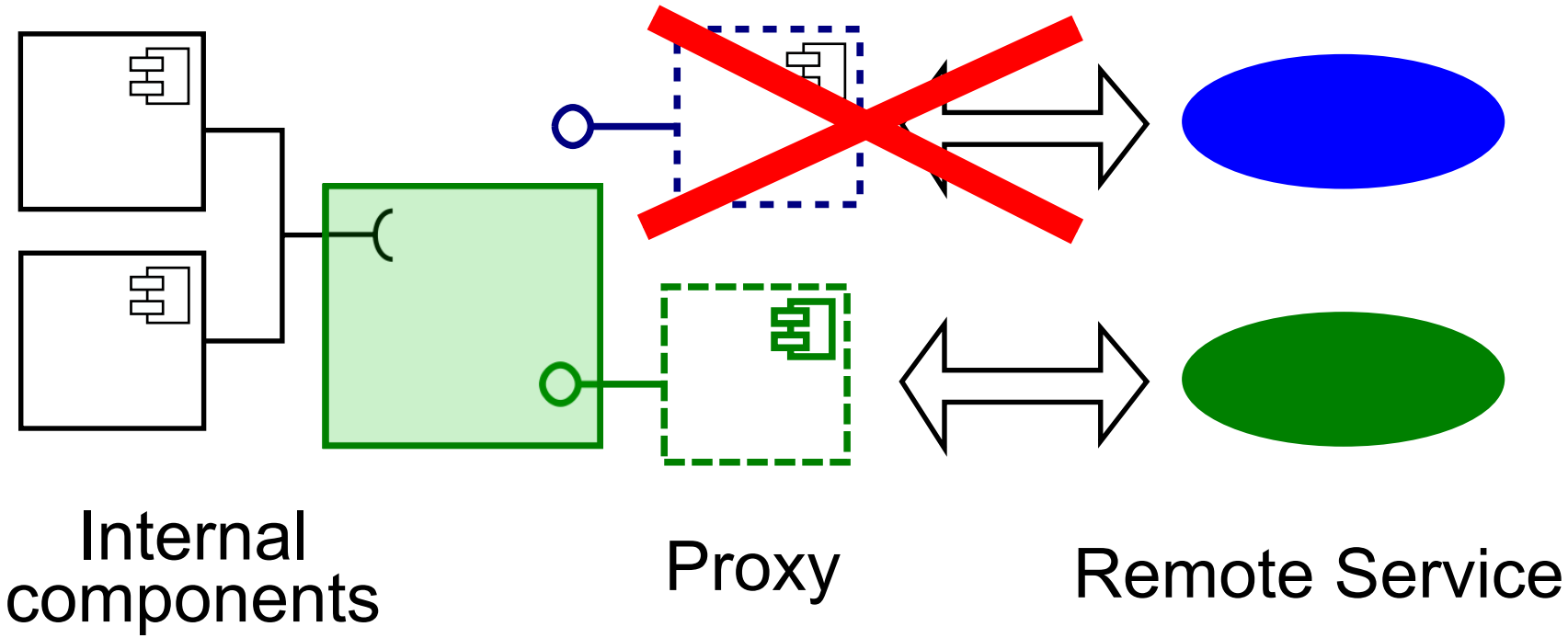
1. Look for an appropriate service
2. Build a local proxy for the remote service
3. Adapt your business objects to the business objects expected/returned by the local proxy

Traditional Applications "Anatomy"



Applications design

Service Replacement Requires To Modify Internal Components



Application design

Traditional Approach Summary

1. Look for an appropriate service
2. Build a local proxy for the remote service
3. Adapt your business objects to the business objects expected/returned by the local proxy

WARNING

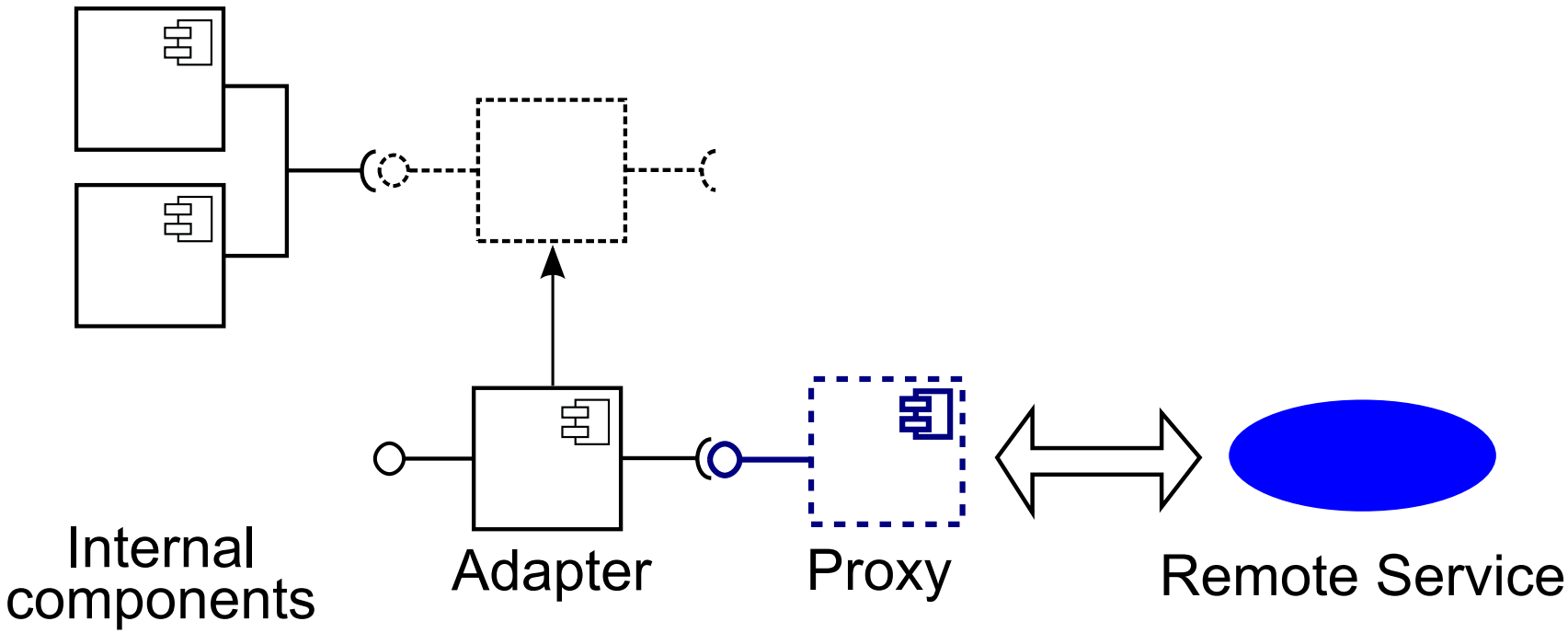
- If the service changes, internal components should be modified
- If it needs to be replaced, internal components should be modified as well

Main Steps for Developing Applications with EasySOC

EasySOC is a novel development model for SOC that improves service replacement [2]

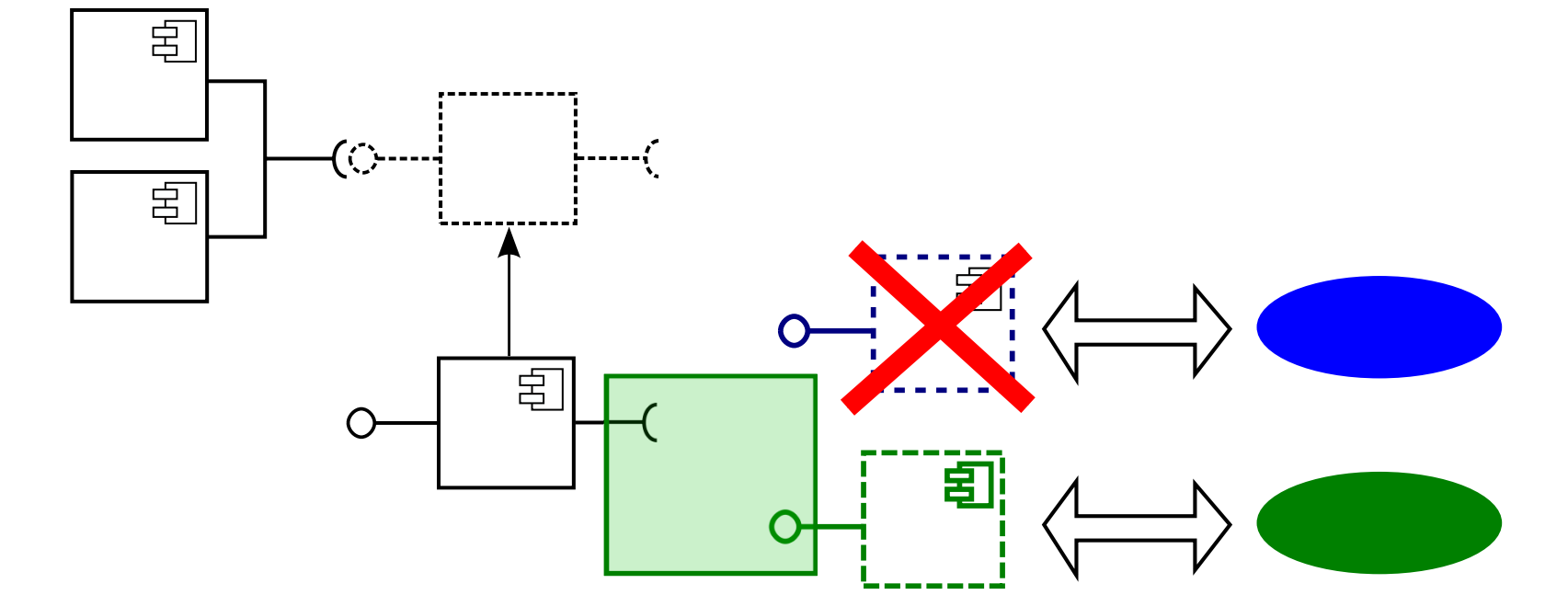
1. Define the expected interface of the service
2. Look for an appropriate service
3. Build a local proxy for the remote service
4. Adapt the business objects expected/returned by the local proxy to your business objects

EasySOC-based Applications "Anatomy"



Application design

Service Replacement Is Transparent for Internal Components



Application design

EasySOC Approach Summary

1. Define the expected interface of the service
2. Look for an appropriate service
3. Build a local proxy for the remote service
4. Adapt the business objects expected/returned by the local proxy to your business objects

If the service changes or it needs to be replaced, its associated adapter should be modified but not internal components

Pros & Cons of the EasySOC Model

Should I tell my team to start using EasySOC?

Cons

- Requires a radical shift in the way applications are developed by the software industry
- Extra step (e.g. for defining the expected service interface)
- Notions of Adapter and Dependency Injection design patterns

Pros

- Shields applications from specific service interfaces [3]
- Simplifies applications maintenance [2]
- The introduced overhead is minimal (jhat +2% MEM) [1]

Pros & Cons of the EasySOC Model

Should I tell my team to start using EasySOC?

Cons

- Requires a radical shift in the way applications are developed by the software industry
- Extra step (e.g. for defining the expected service interface)
- Notions of Adapter and Dependency Injection design patterns

Pros

- Shields applications from specific service interfaces [3]
- Simplifies applications maintenance [2]
- The introduced overhead is minimal (jhat +2% MEM) [1]

Evaluation of Developer's perception of EasySOC

- "Service-Oriented Computing" course of the UNICEN during 2009 http://www.exa.unicen.edu.ar/~cmateos/cos
- 45 postgraduate and undergraduate students

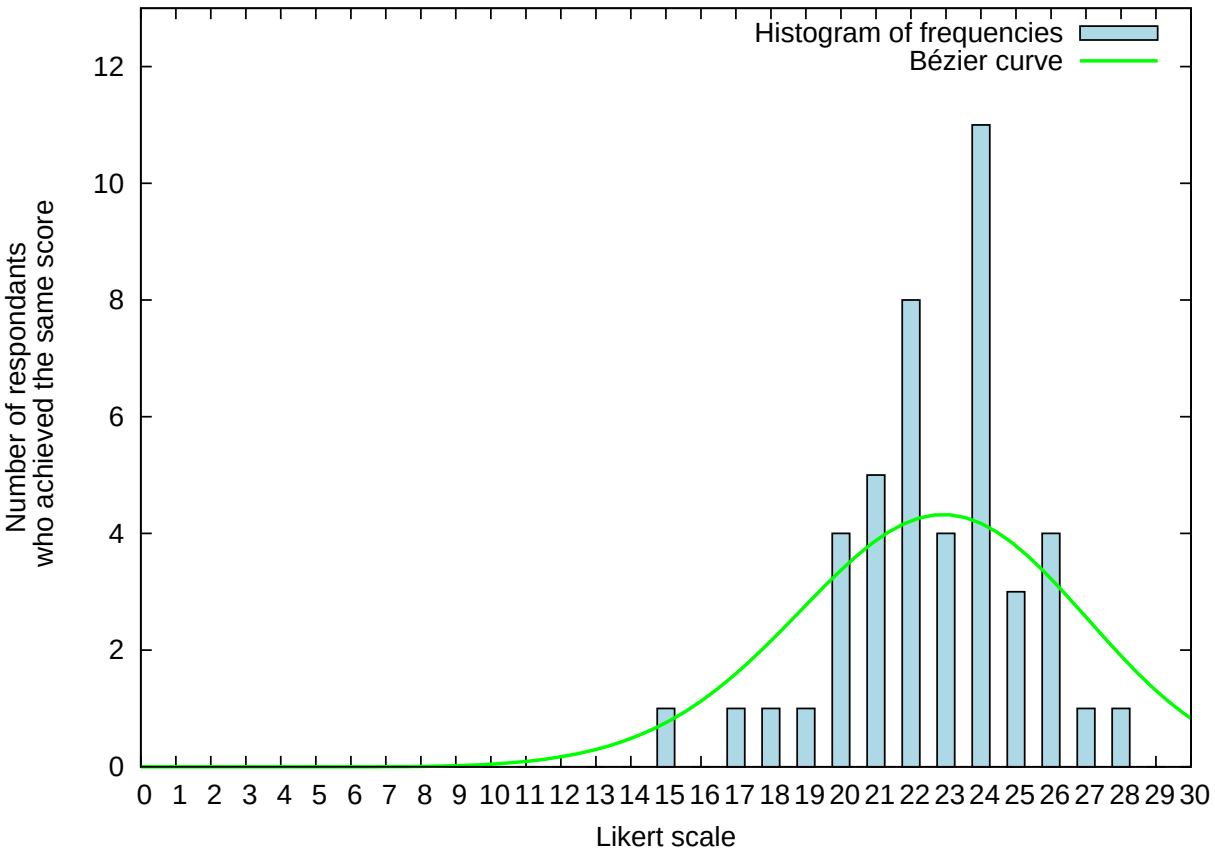
- The participants were asked to design and implement:
  - a Personal Agenda using the Traditional way
  - the same Personal Agenda but using the EasySOC way

- They replaced any service from within both versions afterward
- Finally, they completed a Likert-based questionnaire

The Questionnaire

- Participants can associate a level of agreement with each question:
  - ranging from totally disagree to totally agree
  - no neutral opinion
  - they felt consulted, not evaluated
- To quantify the overall perception of the students on EasySOC:
  - We associated a numerical score with each answer ranging from 0 (totally disagree) to 5 (totally agree)
  - The higher the score the higher the acceptance of EasySOC

Approach preference



- After smoothing:
  - $\mu = 22.67$
  - $\sigma = 2.65$
  - 95.4% scored between $[\mu \pm 2 * \sigma]$

Conclusions

- One of the biggest hurdles in the path of educating students in SOC concepts is the plethora of technologies surrounding the paradigm, which often obscures its cornerstones and eclipses its simplicity
- EasySOC is a novel development model for SOC that improves service replacement
- Experiment results suggest that the analyzed students perceived EasySOC as convenient for implementing applications

Future Work

- What's next?
  - More experiments with 2010 course edition attendants, and real development teams
  - Investigating whether exploiting students' experience in earlier programming paradigms while facilitating activities inherent to SOC and hiding technologies reduces the effort needed to effectively start applying SOC design concepts
- What else is in the paper?
  - a comparison between the traditional model and EasySOC
  - more details about the questionnaire & the participants
  - more discussion about the results
  - students' reasons for choosing EasySOC detailed

For Further Reading I

Marco Crasso, Cristian Mateos, Alejandro Zunino, and Marcelo Campo.
Empirically assessing the impact of dependency injection on the development of Web Service applications.
Journal of Web Engineering, 9(1):66–94, 2010.

Marco Crasso, Cristian Mateos, Alejandro Zunino, and Marcelo Campo.
Easysoc: Making Web Service outsourcing easier.
Information Sciences, Special Issue of Applications of Computational Intelligence and Machine Learning to Software Engineering, in press, acccepted 2010.

For Further Reading II

Cristian Mateos, Marco Crasso, Alejandro Zunino, and Marcelo Campo.

Separation of concerns in service-oriented applications based on pervasive design patterns.

In *SAC-WT'10*, pages 2509–2513. ACM, 2010.