coding {the} architecture

Software Architecture

vs Code

Simon Brown
@simonbrown

“…the architecture diagrams don’t match the code”

Kristijan | Криштиџн
@Krishtidzn

Follow

Any recommendations for software for drawing software architecture but not MS Visio?

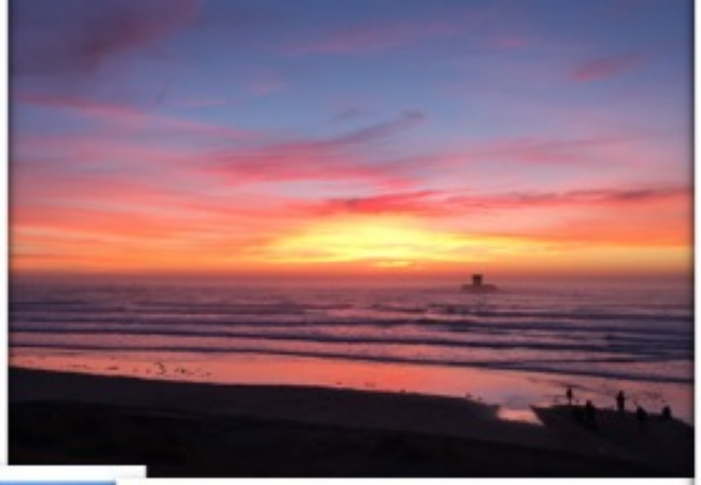
11:11 AM - 16 Apr 2015

Greenland
Iceland
Norwegian Sea
Sweden
Finland
Norway
Denmark
United Kingdom
Ireland
Poland
Germany
Belarus
Jersey
Austria
France
Romania
Italy
Spain
Portugal
Greece
Tunisia
Morocco
Algeria
Libya
Western
North Atlantic Ocean
NL
QC
NB
PE
NS
ME
NH
MA
NY
PA
WV
DE
VA
NC
SC
FL

Jersey

Guernsey

Jersey












I help software teams understand

software architecture, technical leadership and the balance with agility

Software architecture needs to be more accessible

coding {the} architecture

Software Architecture for Developers

Technical leadership by **coding**, coaching, collaboration, *architecture sketching* and just enough up front design

Simon Brown

Leanpub

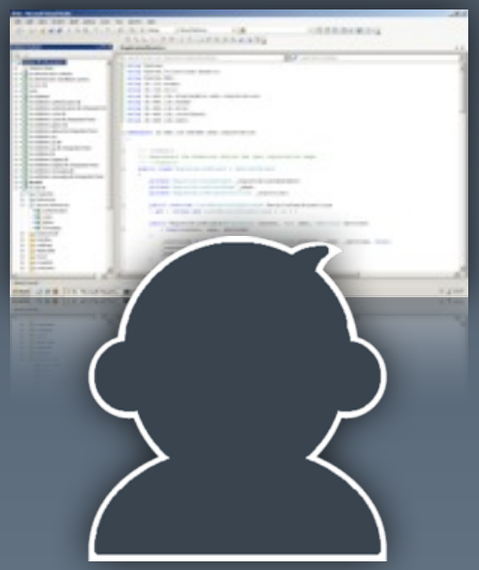
coding {the} architecture

The Art of Visualising Software Architecture

Communicating software architecture with *sketches*, diagrams and the C4 model

Simon Brown

Free!



I code too

⇧ ; - ⇧ 0

The intersection between

software architecture

and

code

The tension between software architecture and code

How do we

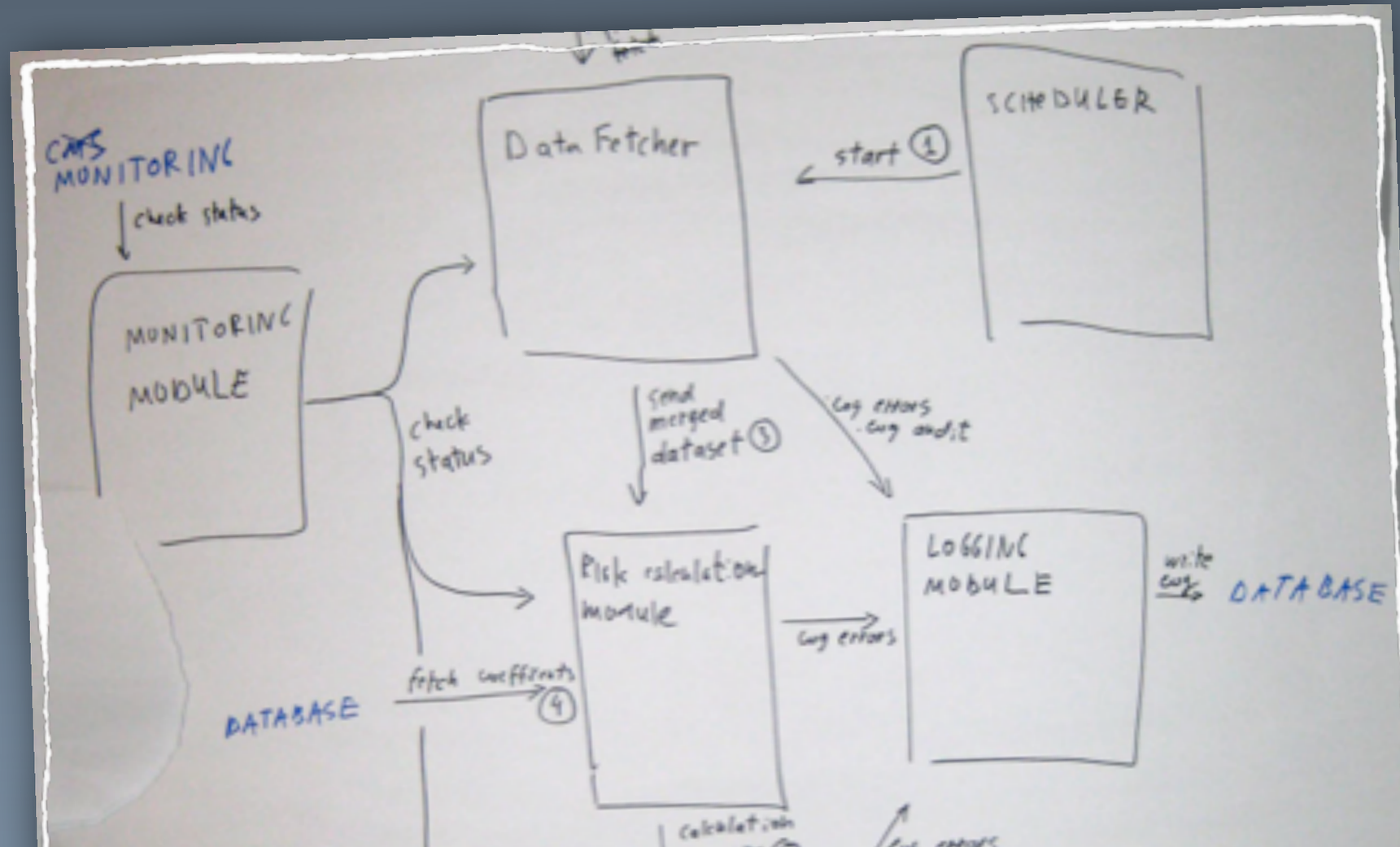
communicate

software architecture?

Abstraction

is about reducing detail

rather than creating a different representation



Abstractions help us

reason about

a big and/or complex
software system

Software architecture deals with abstraction, with decomposition and composition, with style and esthetics.

To describe a software architecture, we use a model composed of multiple views or perspectives.

Architectural Blueprints—The “4+1” View Model of Software Architecture

by Philippe Kruchten

The description of an architecture—the decisions made—can be organized around these four views, and then illustrated by a few selected *use cases*, or *scenarios* which become a fifth view. The architecture is in fact partially evolved from these scenarios as we will see later.

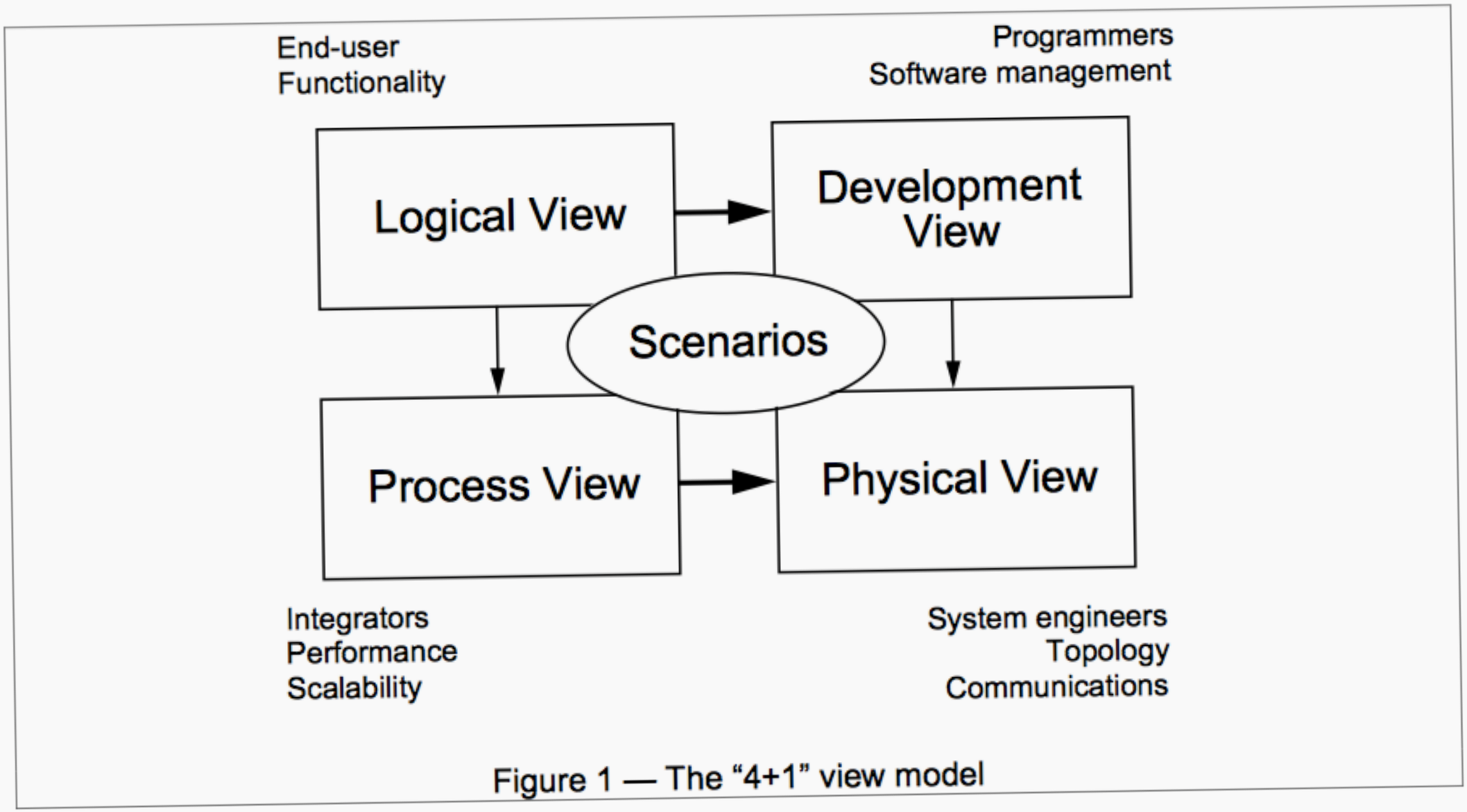
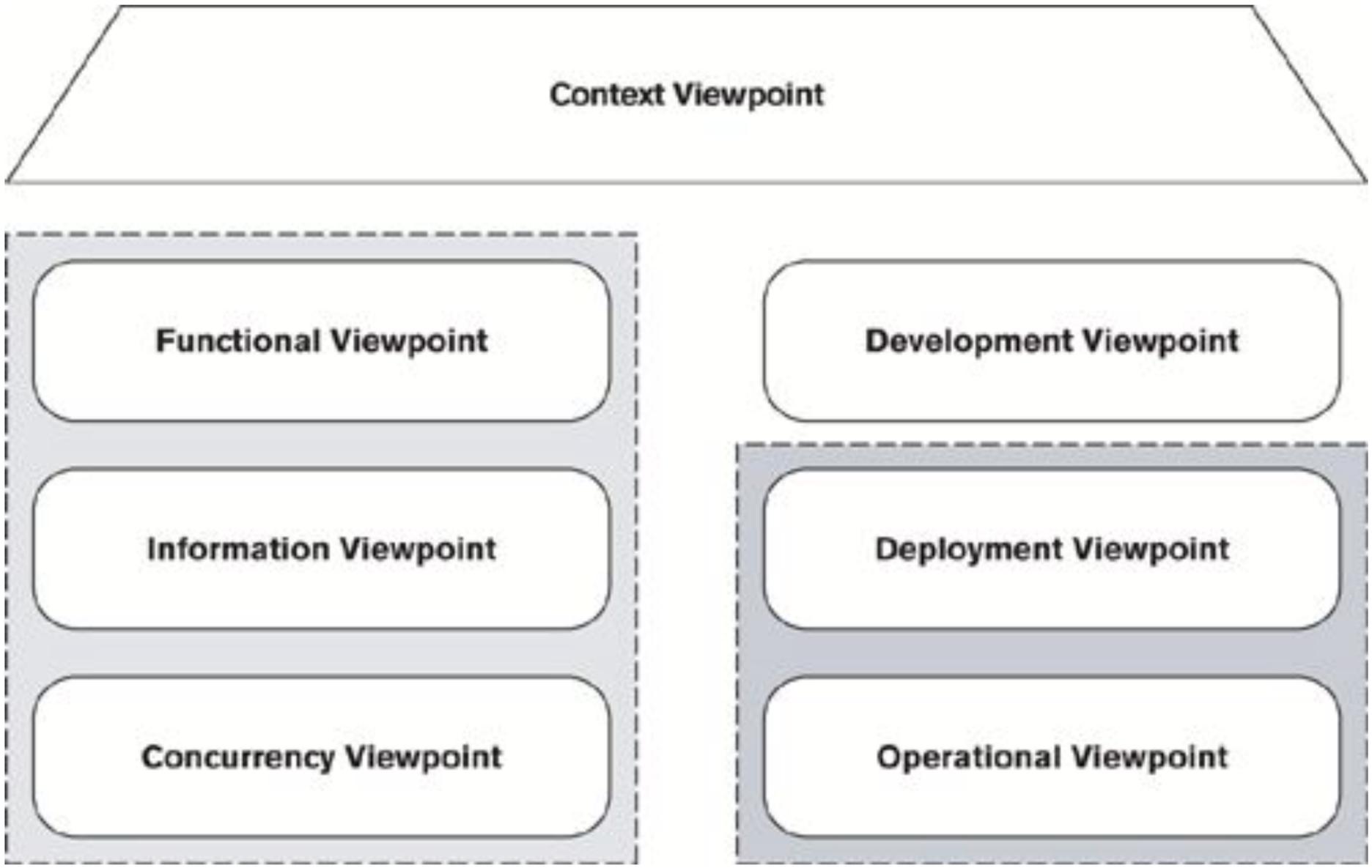
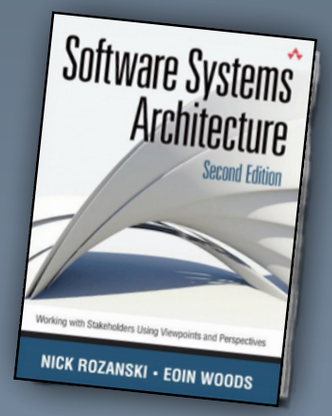


Figure 1 — The "4+1" view model

We apply Perry & Wolf's equation independently on each view, i.e., for each view we define the set of elements to use (components, containers, and connectors) , we capture the forms and patterns that work, and we capture the rationale and constraints, connecting the architecture to some of the requirements.



Viewpoints and perspectives



Viewpoint	Definition
Logical	The logical representation of the system's functional structure, normally presumed to be a class model (in an object-oriented systems development context). Our Functional viewpoint is a development of this "4+1" viewpoint, renamed to make its content clear (because you could have a number of logical aspects to an architecture).
Process	The concurrency and synchronization aspects of the architecture. Our Concurrency viewpoint is a development of this "4+1" viewpoint, renamed to avoid confusion with business process modeling.
Development	The design-time software structure, identifying modules, subsystems, and layers and the concerns directly related to software development. Our Development viewpoint is based on this "4+1" viewpoint.
Physical	The identification of the nodes on which the system's software will be executed and the mapping of other architectural elements to these nodes. Our Deployment viewpoint is a development of this "4+1" viewpoint.

Do the **names**
of those views make sense?

Conceptual vs Logical

Process vs Functional

Development vs Physical

Development vs Implementation

Physical vs Implementation

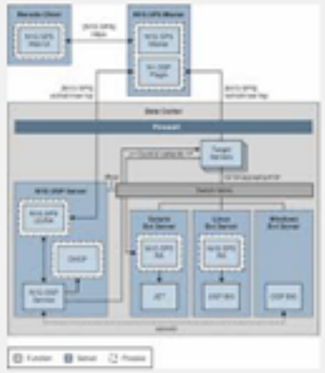
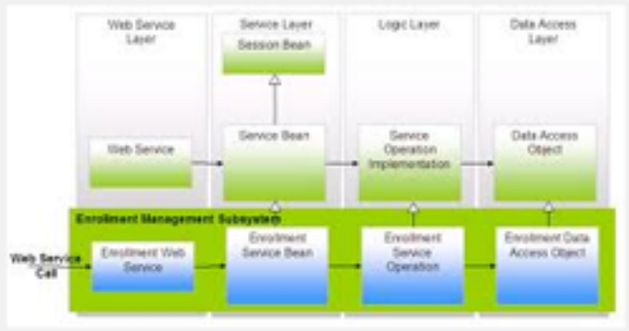
Physical vs Deployment

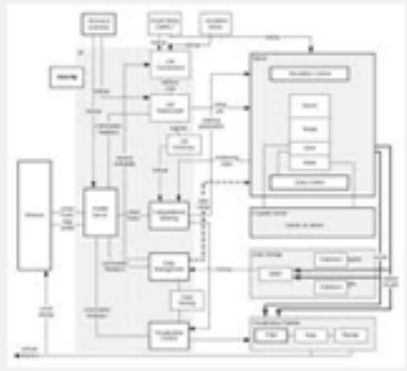
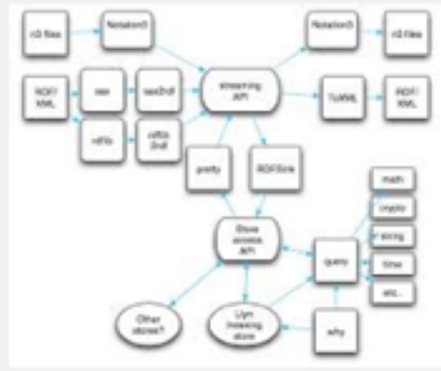
Logical and development views are often separated

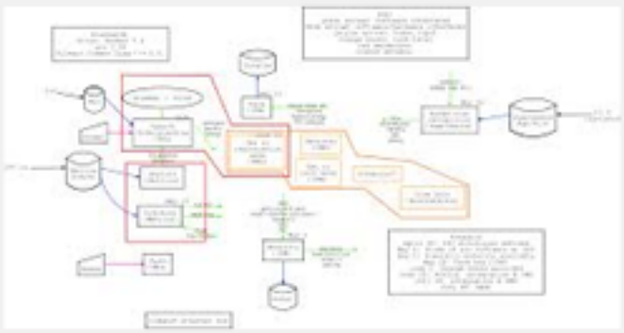
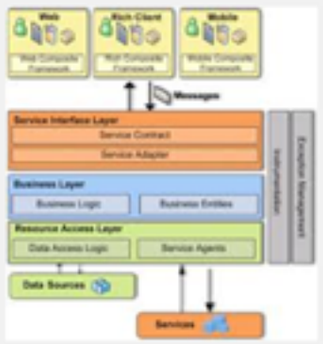
Google software architecture diagram

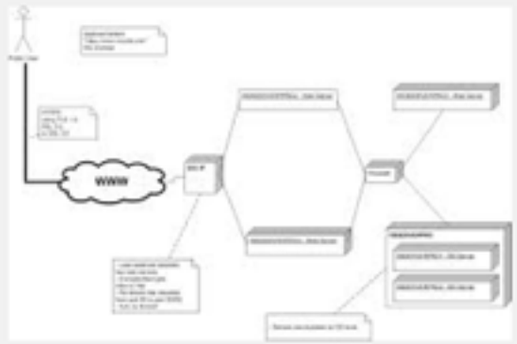
Sign in

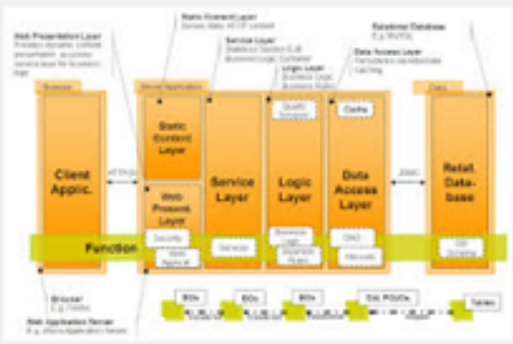
Web Images Videos Shopping News More Search tools SafeSearch

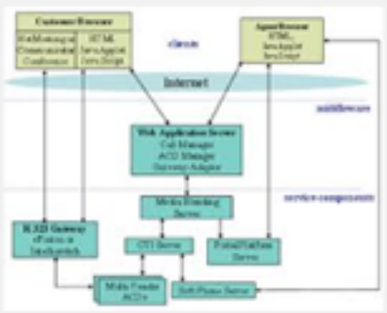




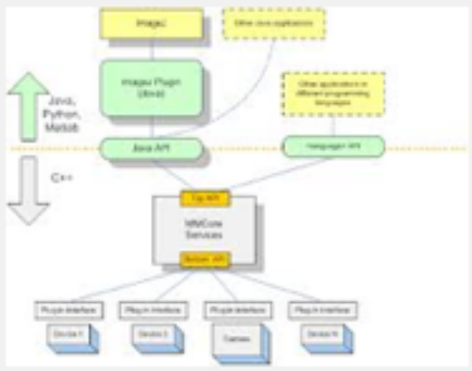



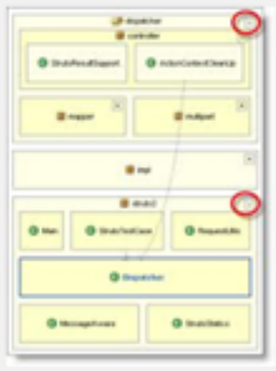



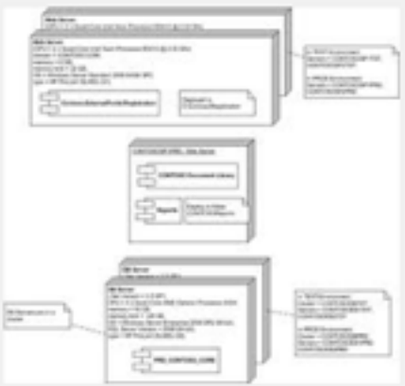
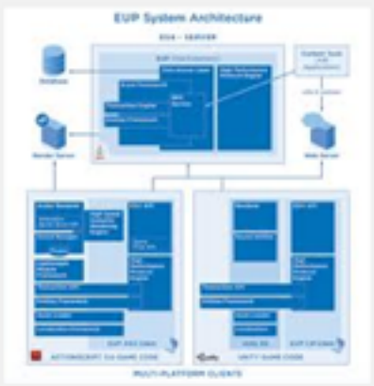






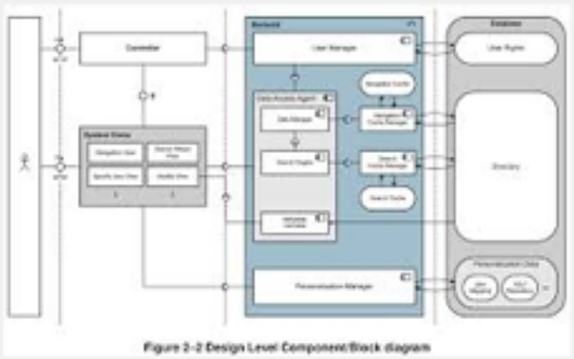
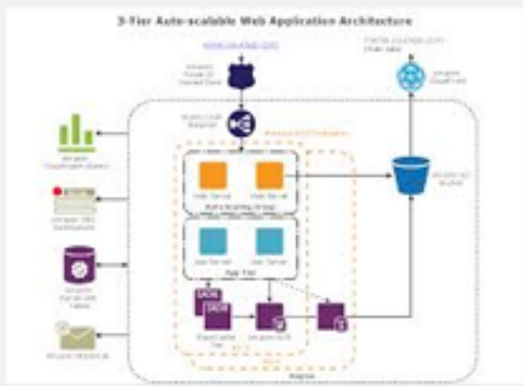


Figure 2-2 Design Level Component/Block diagram


Video over IP
Capturing video frames from Camera
Format conversion and Pre-processing
H.264 Encoder
NALU Fragmentation
Audio/Video sync module
Format conversion and Post-processing
H.264 Decoder
Reconstruction of NALU
RTP/ RTCP
Voice over IP
Acoustic echo cancellation (AEC)
LEC (G.168/ G.165)
AGC - Gain Control
VAD
Speech Encoder
Audio Interface layer
Sound card Interface (PCI / USB)
FXS/TDM Interface
Telephony Interface layer - Tone detection/generation, CPT/DTMF, MF-R1/R2, and Voice Fax Modem Discrimination Unit
Administration UI
Reservation Manager UI
AutoShell
Packaging
Process Automation
Configuration Management / Compliance
Help Desk
Reporting (Jasper)
Agent Remote Deployment
Service Response Monitoring
CA IT Client Manager
CA Patch Manager
CA Process Automation
CA Configuration Automation
CA Service Desk
Chargeback
Agent Policy Configuration
Server Monitoring (Local & Remote)
Web Services and Reliable Transport (Active MQ)
Storage
Platform Management
Microsoft HyperV
Provisioning
Rapid Server Imaging
Automation Infrastructure Platform
AOM Data Service
Policy, Performance, and Resource Management
Authentication / Security (CA EEM)
State and Event Management
Scheduler (Windows)
Discovery (CA Network Discovery Gateway)
Management DB
Performance DB
Service Controller
Rich UI
Web UI
Controls
Service Interface
Activity
Business Rules
Human Workflow
Workflow
Service Agents
DAL
E-Publish
EAI
ECM
DW
OLTP
Signing
Authentication
Authorization
Monitoring
Log & Trace
Exception Management
Configuration
Typical generic software architecture of SCADA systems
Project Editing Service
Computational Analysis Service
e-AIRSmesh
e-AIRSview
Mesh Converter
CFD Solver
Output Converter
Remote Experiment Service
Exp. Exec.
Exp. Info.
Aerodynamic Exp.
PIV Exp.
Parametric Study
Validation
Analysis
Collaboration
Session Reservation
Resource Configuration Service
Resource Monitoring Service
ASP (Scientific Application Service Provider)
Basic Execution Service
Parameter Sweeper
Global Scheduler
Resource Catalog
GRAM adapter
Plotting Service
eAIRS Plot
Conv. Graph
Data Transfer Service
Metadata Management Service
AG + VNC
Visualization Sharing
Mobile Service Component
A/V Conference
Chat
Application
Output device manager
Output devices
User manager
DB-Server
Analysis

Would we

code

it that way?

Did we

code

it that way?

As an industry, we lack a common vocabulary with which to think about, describe and communicate software architecture

Who here uses UML

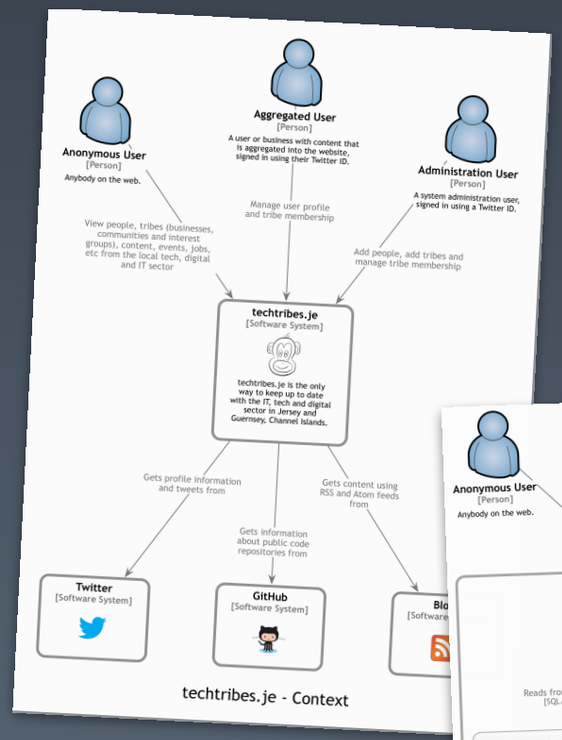
on a regular basis?

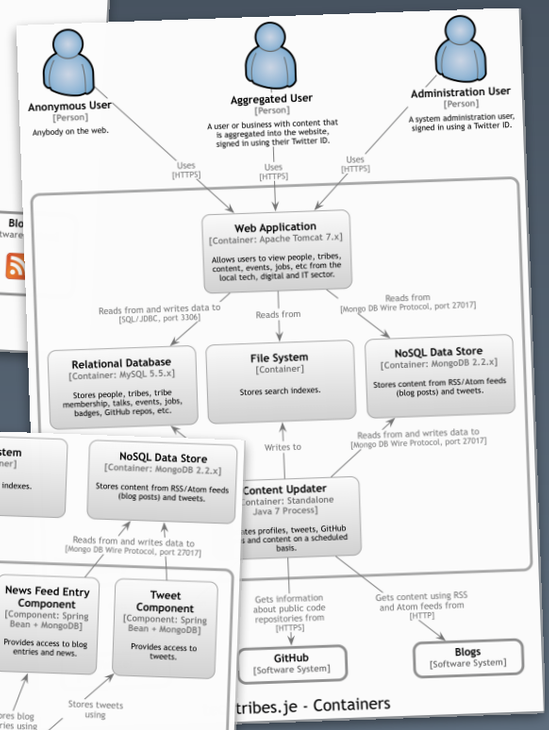
9 out of 10 people don't use UML

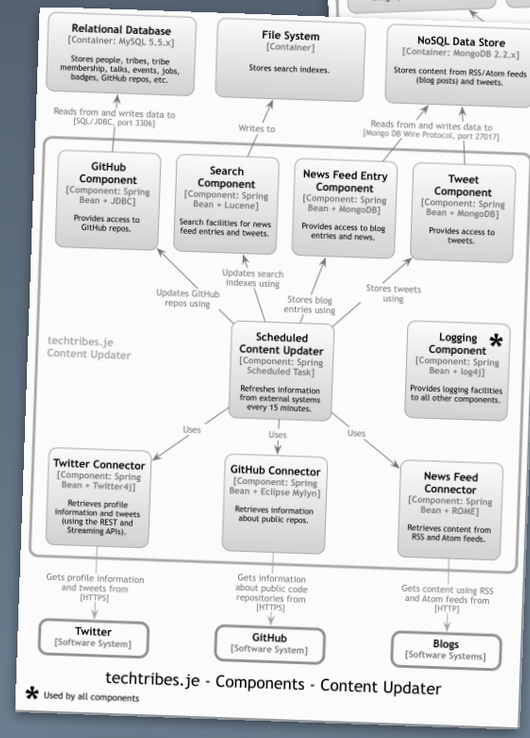
(in my experience)

A common set of abstractions

is more important than
a common notation

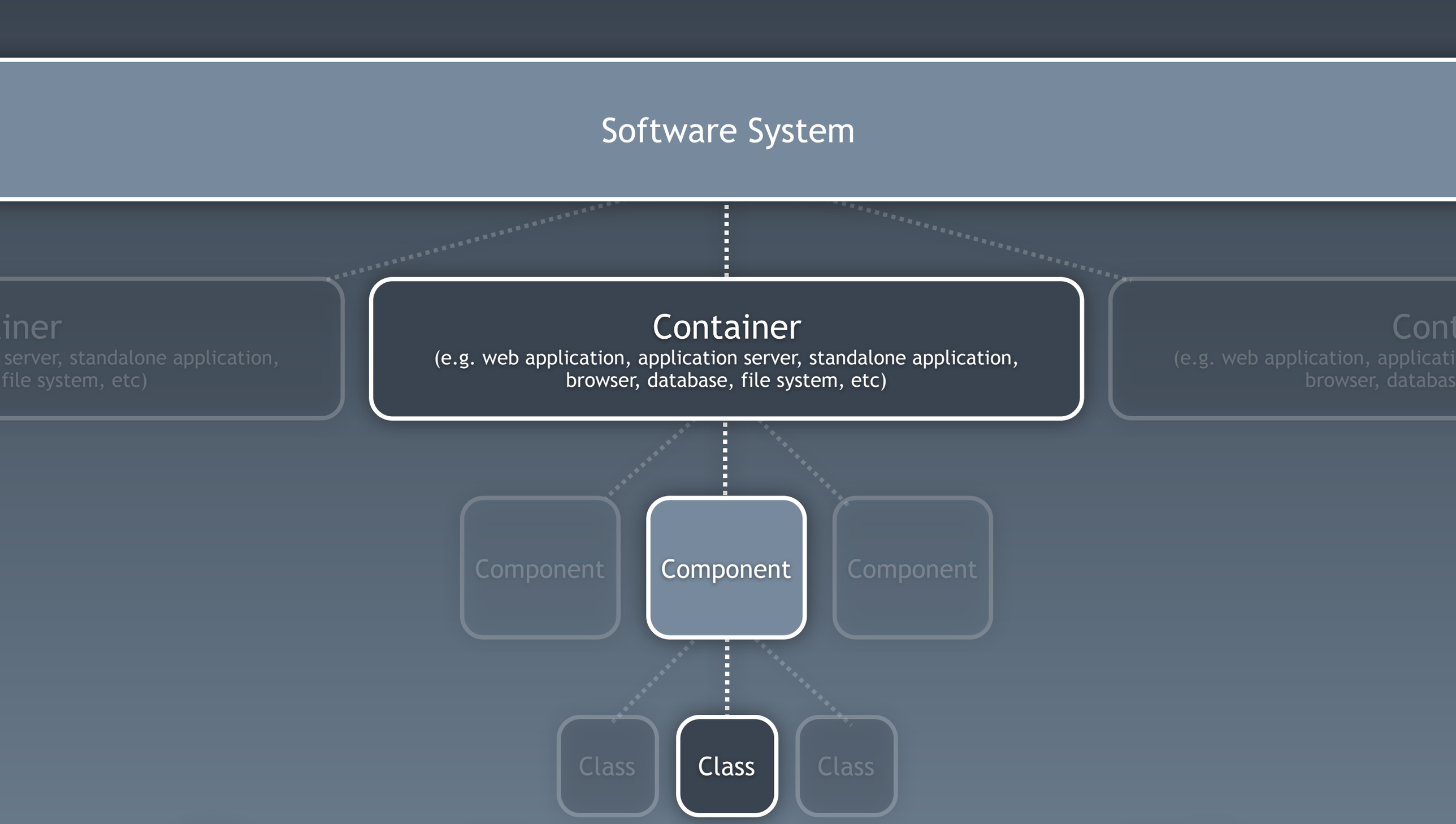






Diagrams are maps

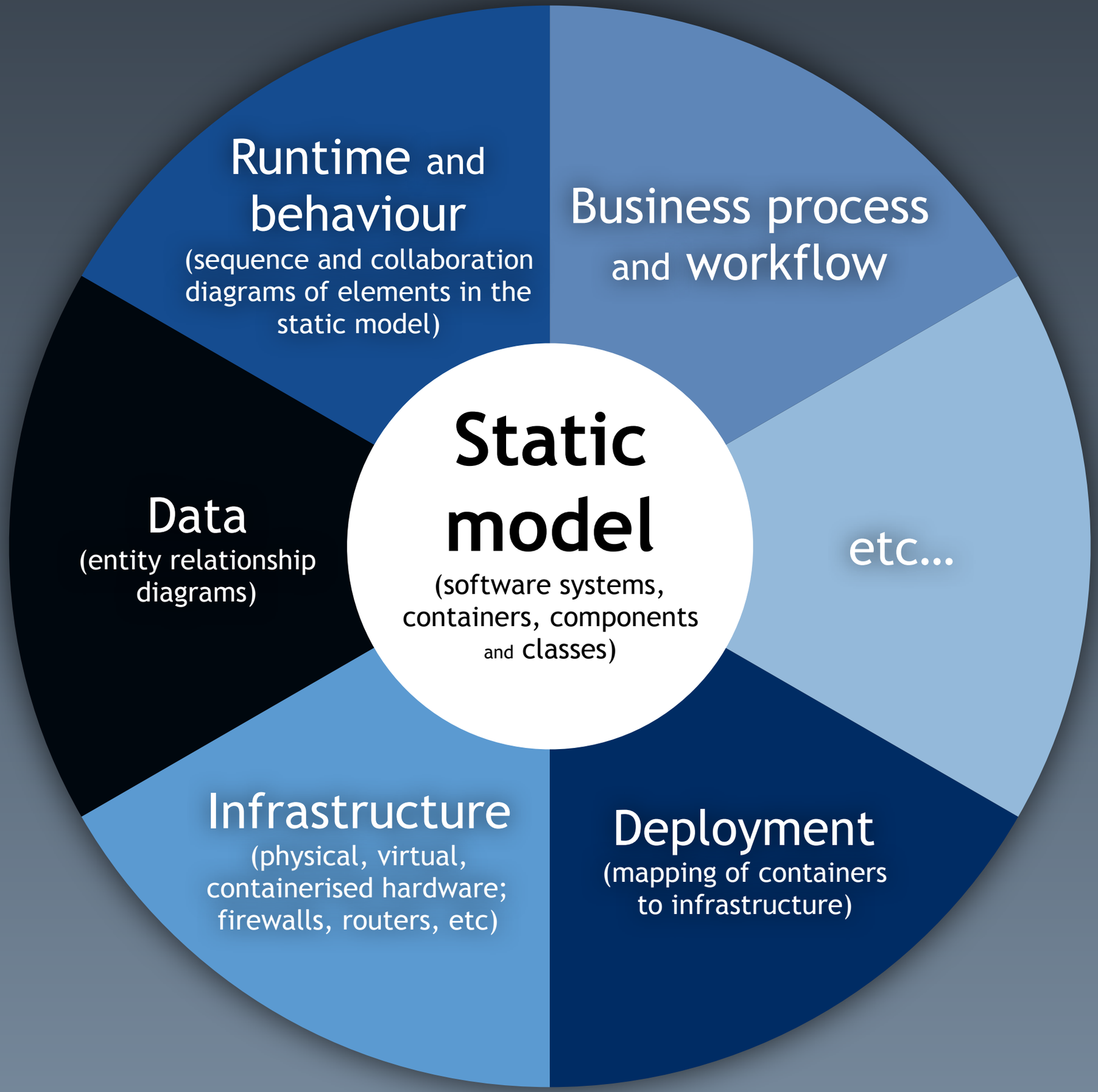
that help a team navigate a complex codebase



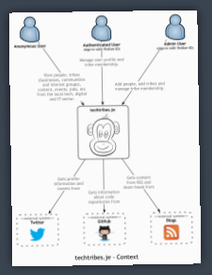
A **software system** is made up of one or more **containers**,

each of which contains one or more **components**,

which in turn are implemented by one or more **classes**.

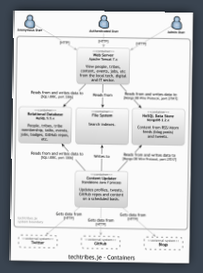

Runtime and behaviour
(sequence and collaboration diagrams of elements in the static model)
Business process and workflow
Static model
(software systems, containers, components and classes)
etc...
Data
(entity relationship diagrams)
Infrastructure
(physical, virtual, containerised hardware; firewalls, routers, etc)
Deployment
(mapping of containers to infrastructure)

The C4 model



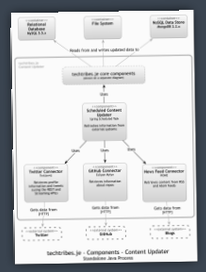
System Context

The system plus users and system dependencies



Containers

The overall shape of the architecture and technology choices



Components

Logical components and their interactions within a container

Classes

Component or pattern implementation details

C4++

Enterprise context

User interface mockups and wireframes

Domain model

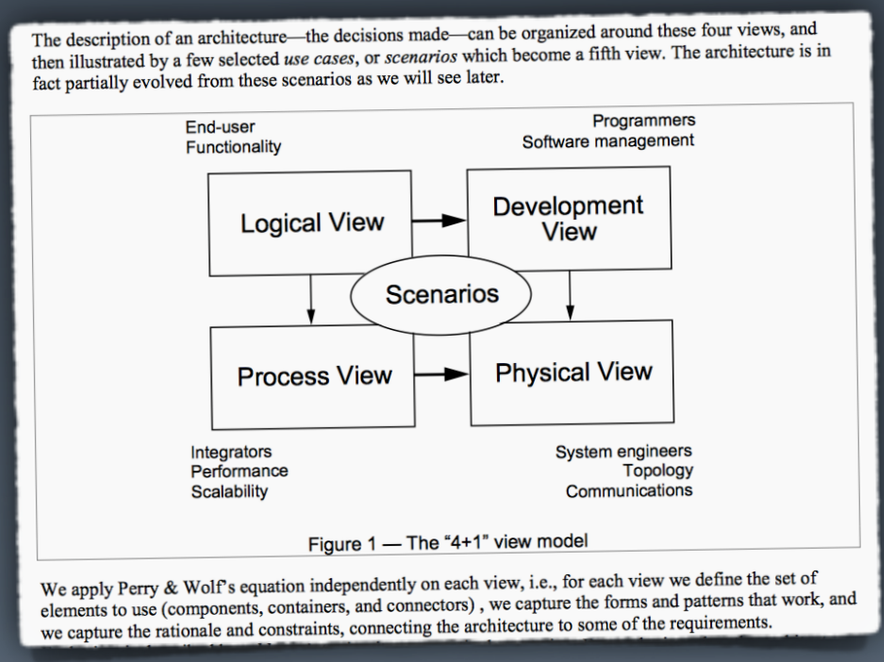
Sequence and collaboration diagrams

Business process and workflow models

Infrastructure model

Deployment model

...



The description of an architecture—the decisions made—can be organized around these four views, and then illustrated by a few selected *use cases*, or *scenarios* which become a fifth view. The architecture is in fact partially evolved from these scenarios as we will see later.

Figure 1 — The "4+1" view model

We apply Perry & Wolf's equation independently on each view, i.e., for each view we define the set of elements to use (components, containers, and connectors) , we capture the forms and patterns that work, and we capture the rationale and constraints, connecting the architecture to some of the requirements.

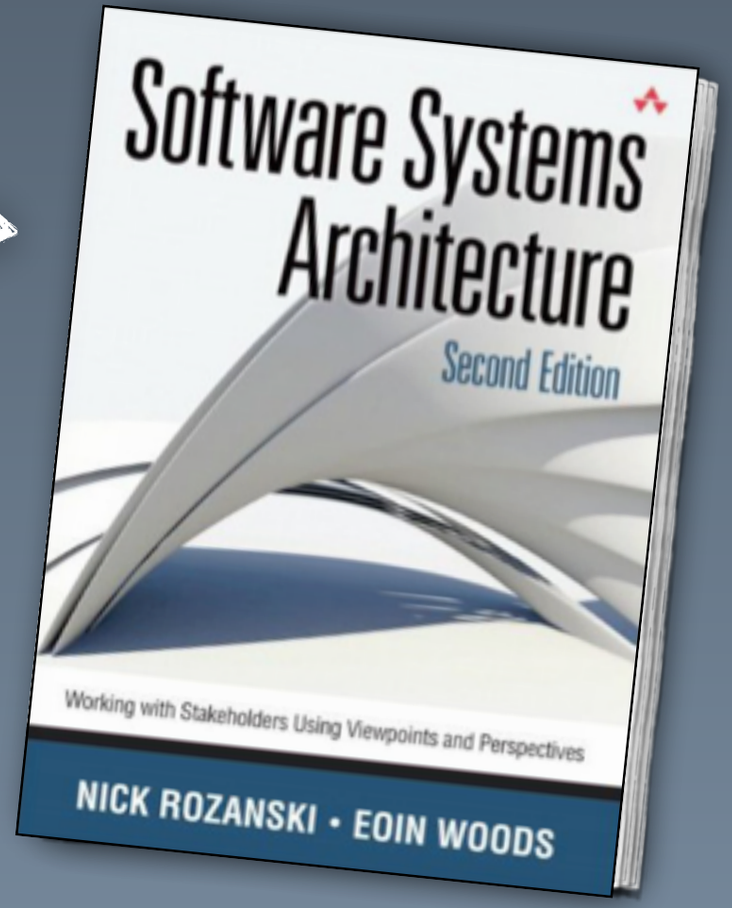
4+1 architectural view model

Philippe Kruchten

Software Systems Architecture

Working with Stakeholders Using Viewpoints and Perspectives (2nd Edition)

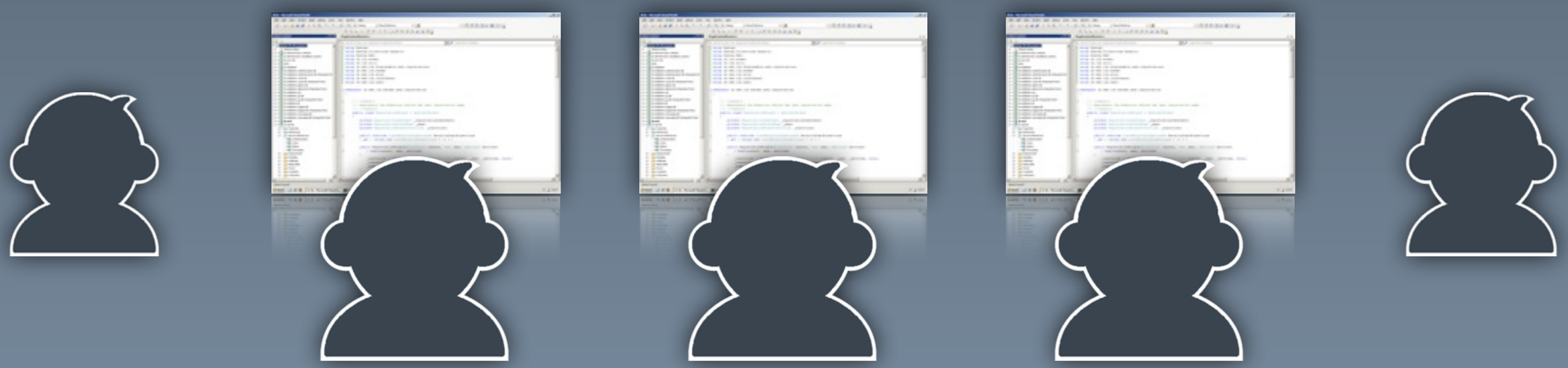
Nick Rozanski and Eoin Woods



Software developers are the most important

stakeholders

of software architecture

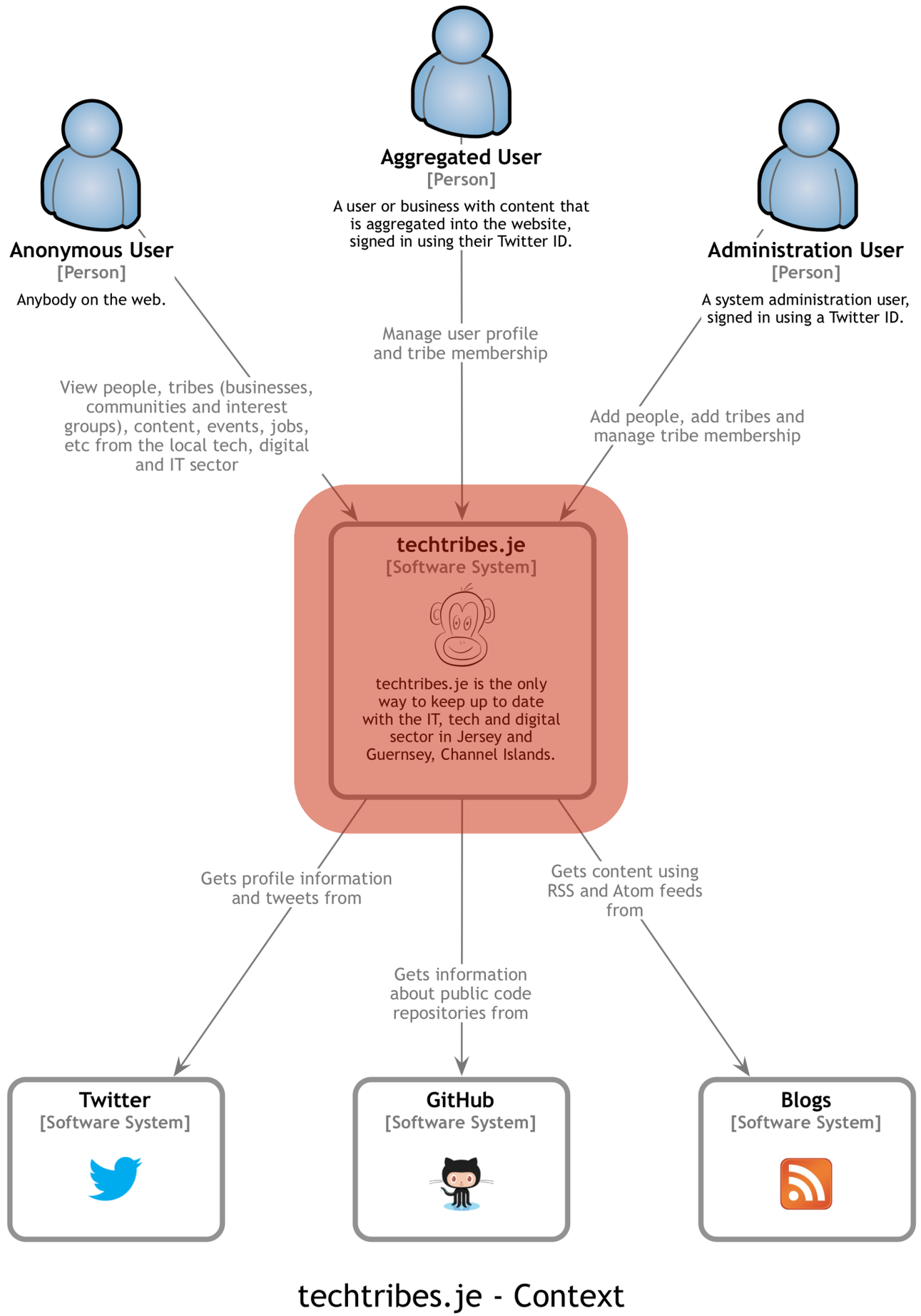


Context diagram
(level 1)

Container diagram
(level 2)

Component diagram
(level 3)

Class diagram
(level 4)



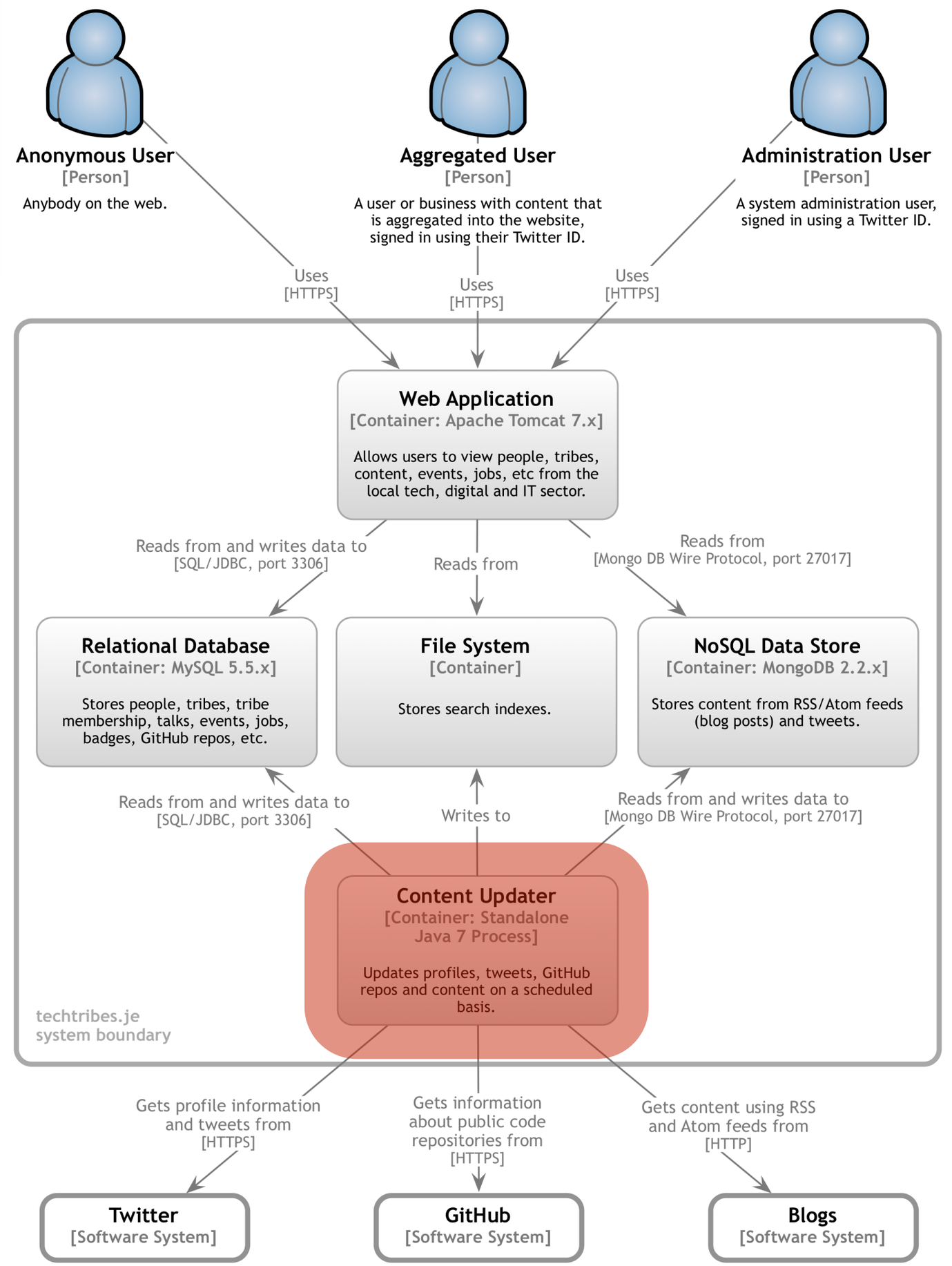
techtribes.je - Context

Context diagram
(level 1)

Container diagram
(level 2)

Component diagram
(level 3)

Class diagram
(level 4)



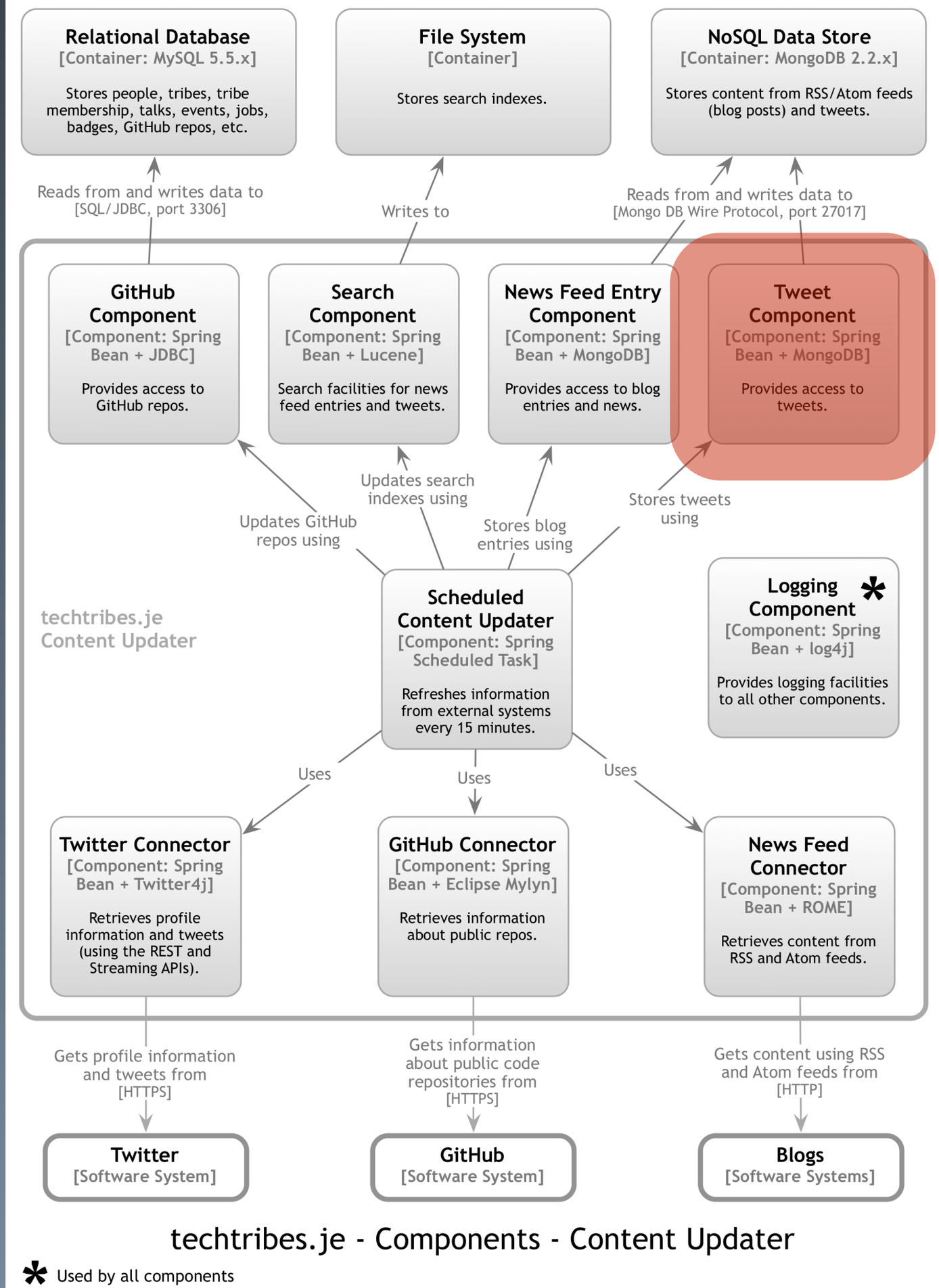
techtribes.je - Containers

Context diagram
(level 1)

Container diagram
(level 2)

Component diagram
(level 3)

Class diagram
(level 4)



techtribes.je - Components - Content Updater

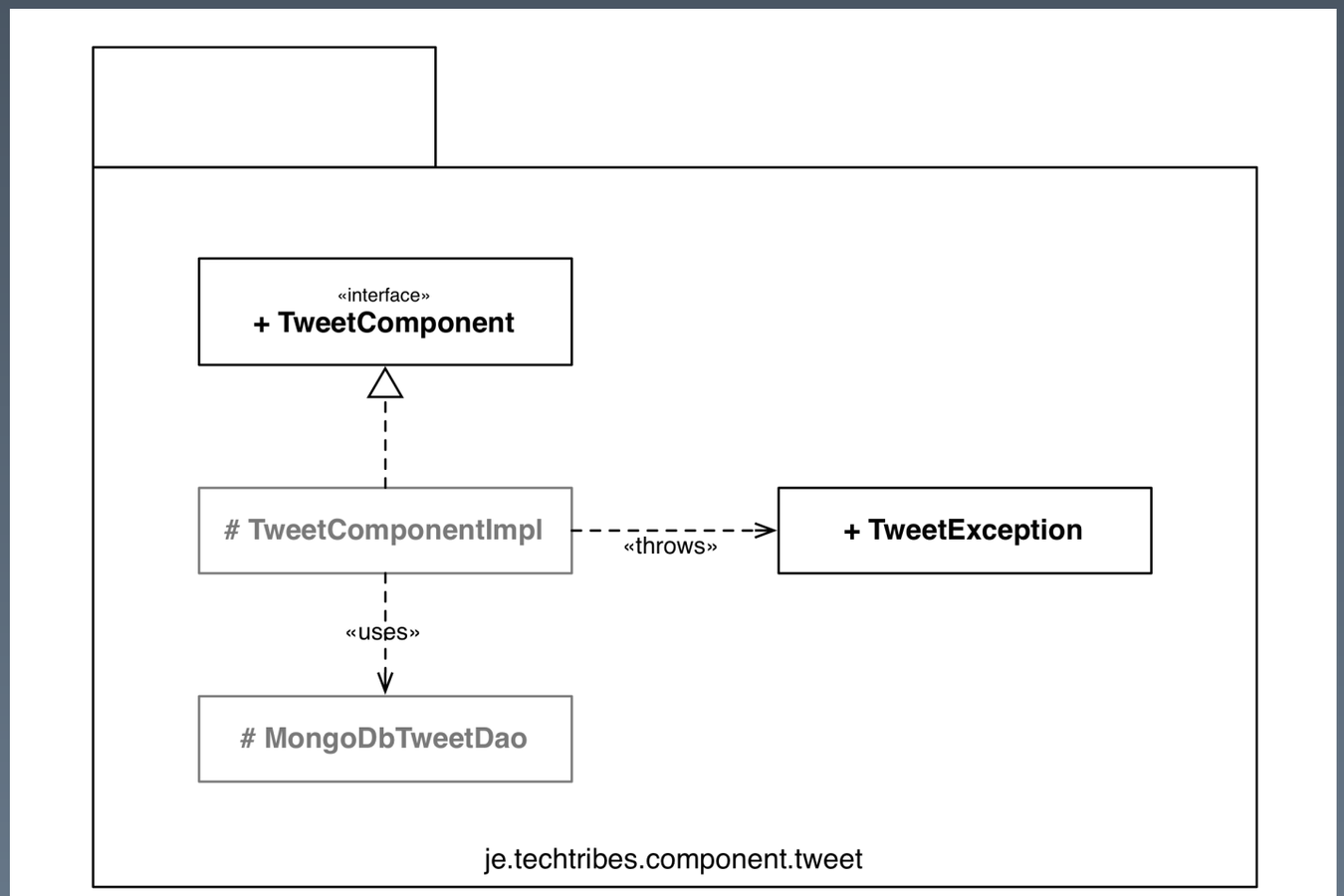
* Used by all components

Context diagram
(level 1)

Container diagram
(level 2)

Component diagram
(level 3)

Class diagram
(level 4)



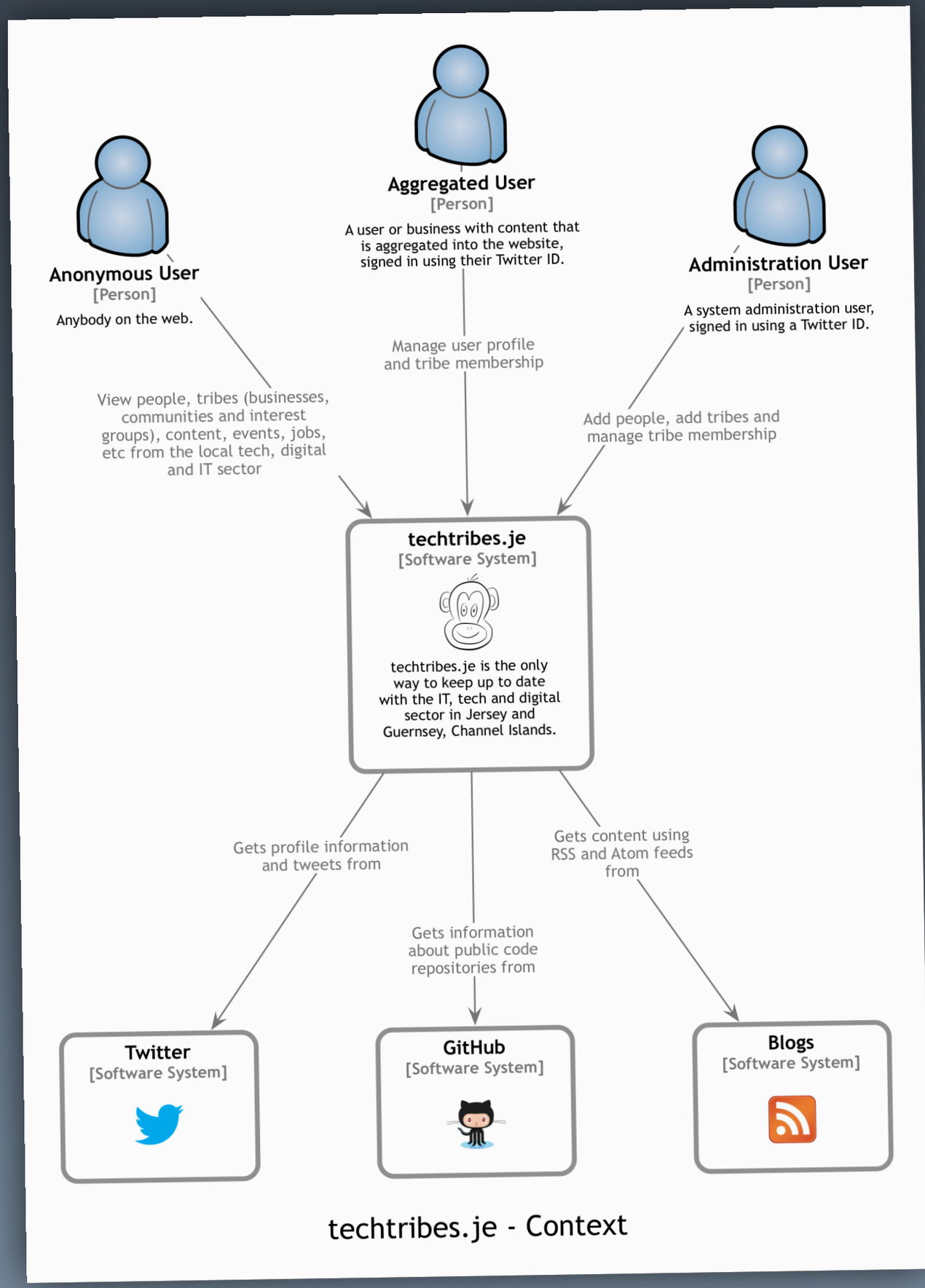
Tools for creating diagrams

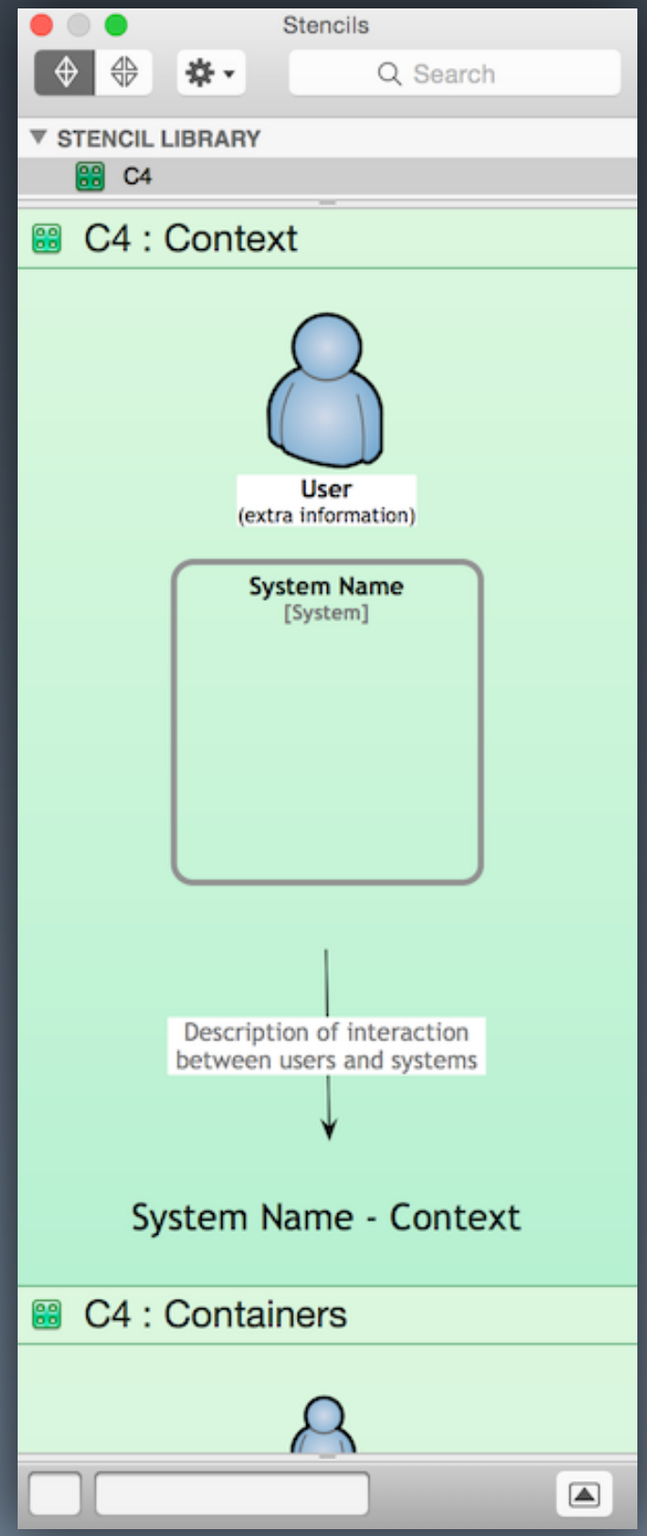
What's the easiest solution to create software architecture diagrams?

I don't want to bother with adhering to UML or finding the proper symbol for the right type of component. I might want a cloud, or 5 boxes in a larger box. Or some other shape. And I'd like a nice variety of arrows to choose from. Something to whip together a diagram fast, but equally important, it should look pretty. Something professional, that could go in a publication or on a company's web site. Certainly free is better, but I think I'm willing to pay once to not have to fight the whatever tool I use to document software anymore.

Any suggestions?

http://quora.com/Whats-the-easiest-solution-to-create-software-architecture-diagrams


Aggregated User
[Person]
A user or business with content that is aggregated into the website, signed in using their Twitter ID.
Anonymous User
[Person]
Anybody on the web.
Administration User
[Person]
A system administration user, signed in using a Twitter ID.
Manage user profile and tribe membership
View people, tribes (businesses, communities and interest groups), content, events, jobs, etc from the local tech, digital and IT sector
Add people, add tribes and manage tribe membership
techtribes.je
[Software System]
techtribes.je is the only way to keep up to date with the IT, tech and digital sector in Jersey and Guernsey, Channel Islands.
Gets profile information and tweets from
Gets information about public code repositories from
Gets content using RSS and Atom feeds from
Twitter
[Software System]
GitHub
[Software System]
Blogs
[Software System]
techtribes.je - Context


Stencils
Search
STENCIL LIBRARY
C4
C4 : Context
User
(extra information)
System Name
[System]
Description of interaction between users and systems
System Name - Context
C4 : Containers

Tools?

It’s 2016, why are
we still using
general purpose
diagramming tools
to draw software architecture
diagrams?

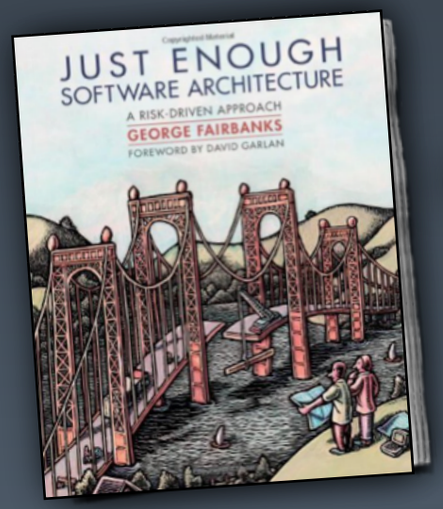
Sketches get out of date,
so why not
auto-generate
the diagrams?

Diagramming tools see

packages

and classes

rather than components



“the model-code gap”

Model-code gap. Your architecture models and your source code will not show the same things. The difference between them is the *model-code gap*. Your architecture models include some abstract concepts, like components, that your programming language does not, but could. Beyond that, architecture models include intensional elements, like design decisions and constraints, that cannot be expressed in procedural source code at all.

Consequently, the relationship between the architecture model and source code is complicated. It is mostly a refinement relationship, where the extensional elements in the architecture model are refined into extensional elements in source code. This is shown in Figure 10.3. However, intensional elements are not refined into corresponding elements in source code.

Upon learning about the model-code gap, your first instinct may be to avoid it. But reflecting on the origins of the gap gives little hope of a general solution in the short term: architecture models help you reason about complexity and scale because they are abstract and intensional; source code executes on machines because it is concrete and extensional.

Software Reflexion Mo
Bridging the Gap between Source and

Gail C. Murphy and David Notkin

Dept. of Computer Science & Engineering
University of Washington
Box 352350
Seattle WA, USA 98195-2350
{gmurphy, notkin}@cs.washington.edu

Abstract

Software engineers often use high-level models (for instance, box and arrow sketches) to reason and communicate about an existing software system. One problem with high-level models is that they are almost always inaccurate with respect to the system's source code. We have developed an approach that helps an engineer use a high-level model of the structure of an existing software system as a lens through which to see a model of that system's source code. In particular, an engineer defines a high-level model and specifies how the model maps to the source. A tool then computes a software reflexion model that shows where the engineer's high-level model agrees with and where it differs from a model of the source.

The paper provides a formal characterization of reflexion models, discusses practical aspects of the approach, and relates experiences of applying the approach and tools to a number of different systems. The illustrative example used in the paper describes the application of reflexion models to NetBSD, an implementation of Unix comprised of 250,000 lines of C code. In only a few hours, an engineer computed several reflexion models that provided him with a useful, global overview of the structure of the NetBSD virtual memory subsystem. The approach has also been applied to aid in the understanding and experimental reengineering of the Microsoft Excel spreadsheet product.

*This research was funded in part by the NSF grant CCR-8858804 and a Canadian NSERC post-graduate scholarship.

SIGSOFT '95 Washington, D.C., USA
©1995 ACM 0-89791-716-2/95/0010...$3.50

1 Intr

Software e
software sy
Box and ar
are often f
though th
soning abo
can be dan
always ina
source.

Current
high-level
derived m
their very
source. Al
by these re
from the m
ple of this

We have
Figure 1,
sufficiently
ent way. T
of interest.
call graph
source cod
between th
model is th
engineer's
agree with
terprets th
modifies th
tional refle

[1]The old
flexion" from

1 Introduction

Software engineers often think about an existing software system in terms of high-level models. Box and arrow sketches of a system, for instance, are often found on engineers' whiteboards. Although these models are commonly used, reasoning about the system in terms of such models can be dangerous because the models are almost always inaccurate with respect to the system's source.

Current reverse engineering systems derive high-level models from the source code. These derived models are useful because they are, by their very nature, accurate representations of the source. Although accurate, the models created by these reverse engineering systems may differ from the models sketched by engineers; an exam-

The intersection of
software architecture
and code

Does your code reflect the

abstractions

that you think about?

Abstractions

on diagrams

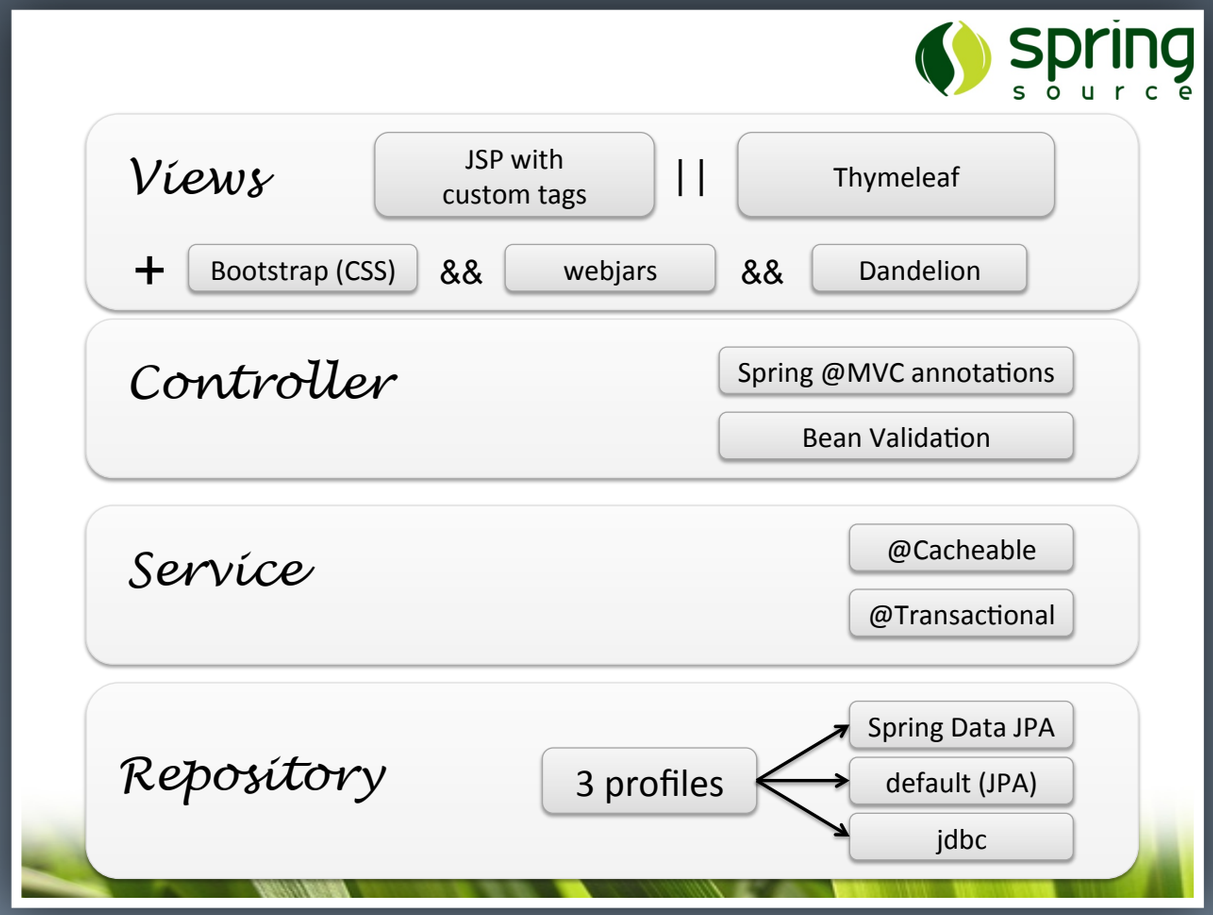
should reflect the

code

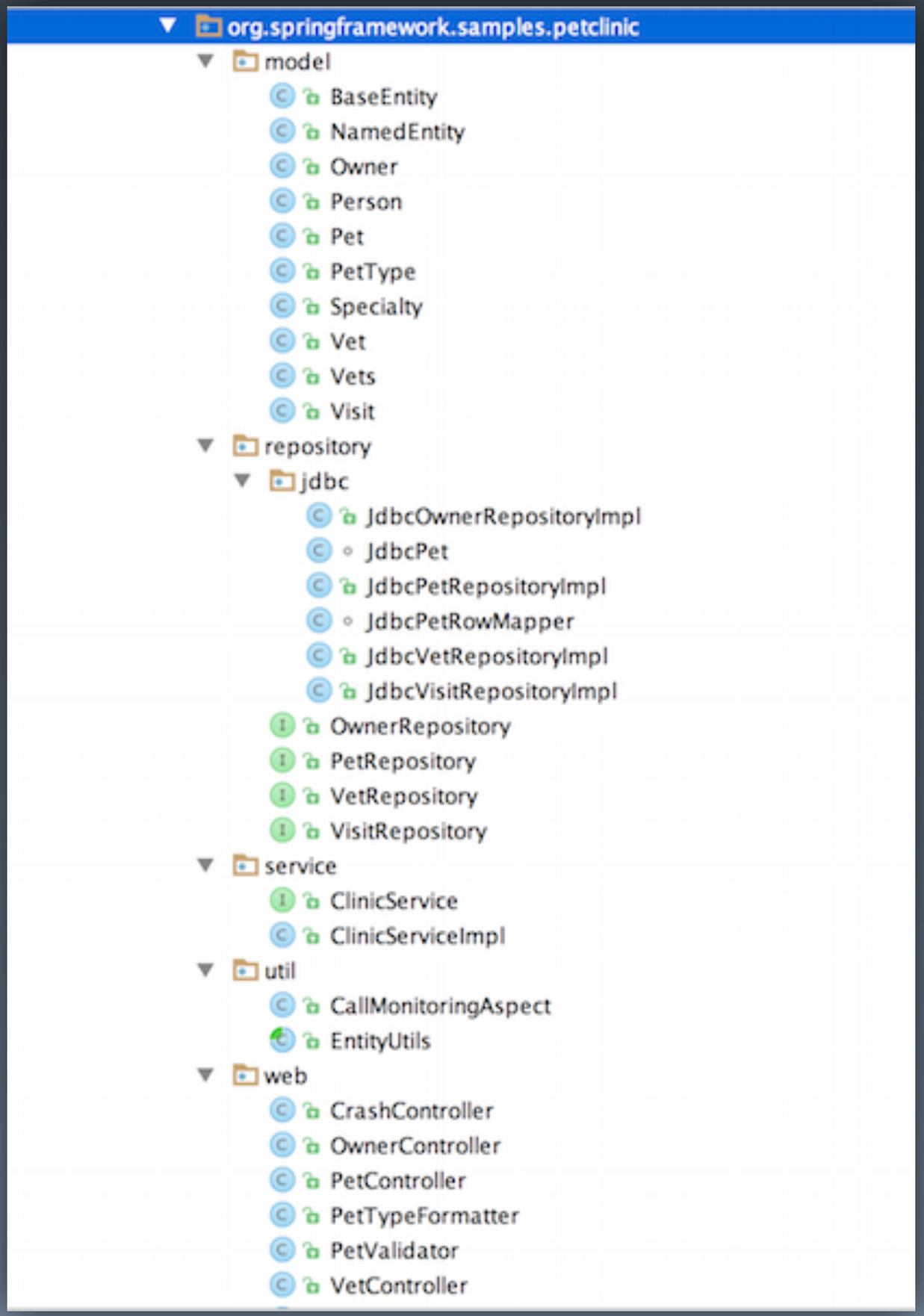
What is a “component”?

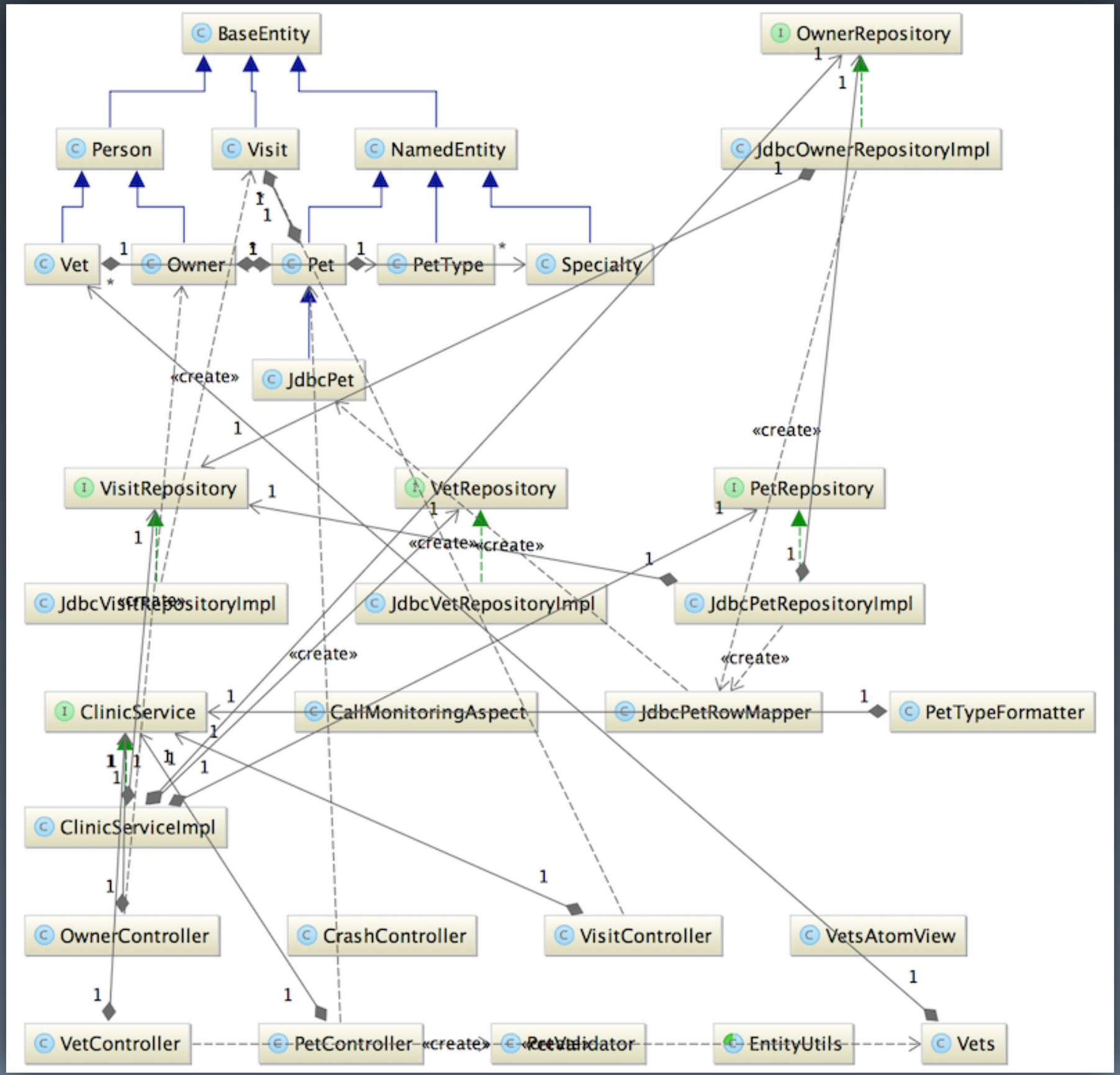
Spring PetClinic

https://github.com/spring-projects/spring-petclinic/



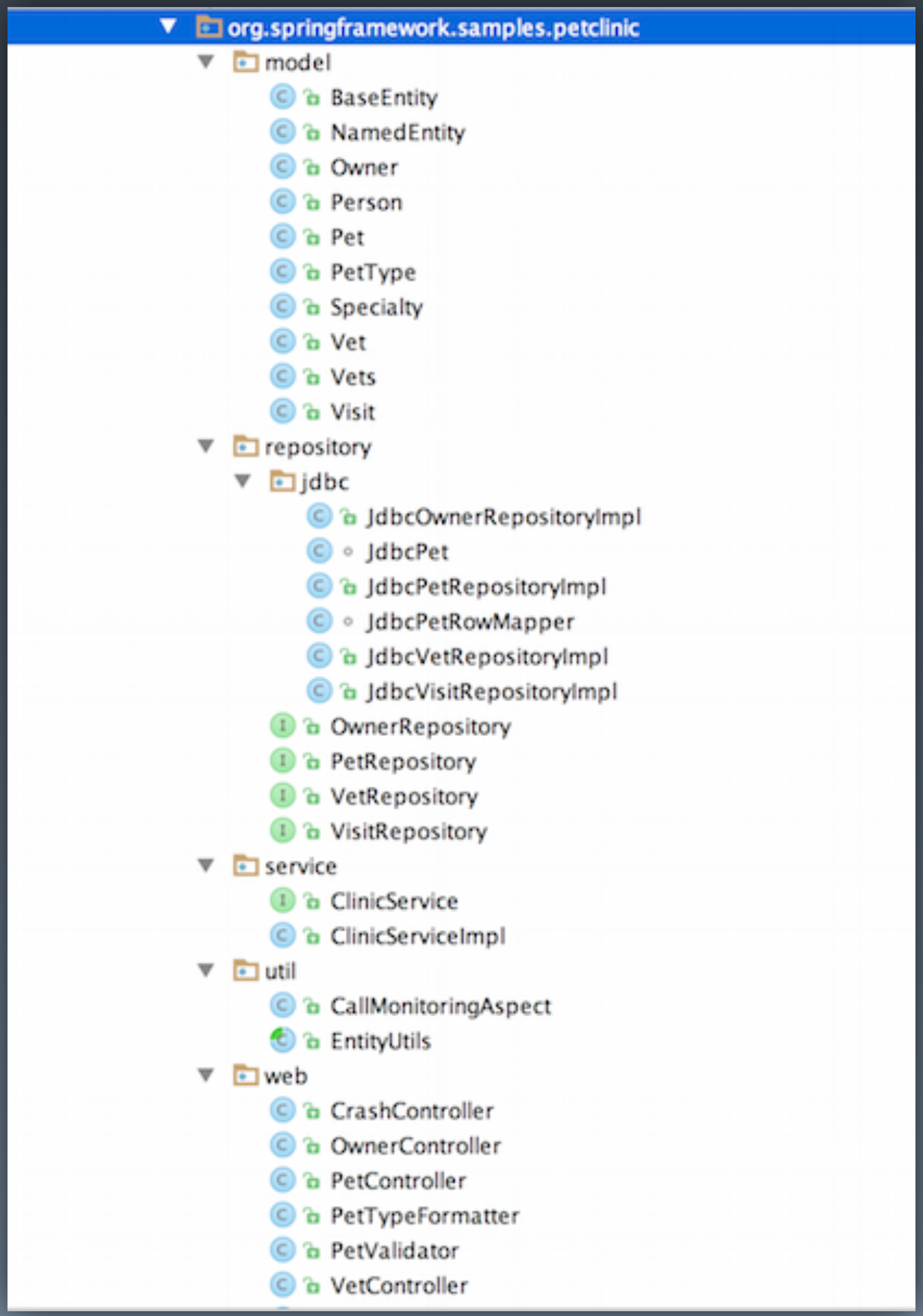
https://speakerdeck.com/michaelisvy/spring-petclinic-sample-application

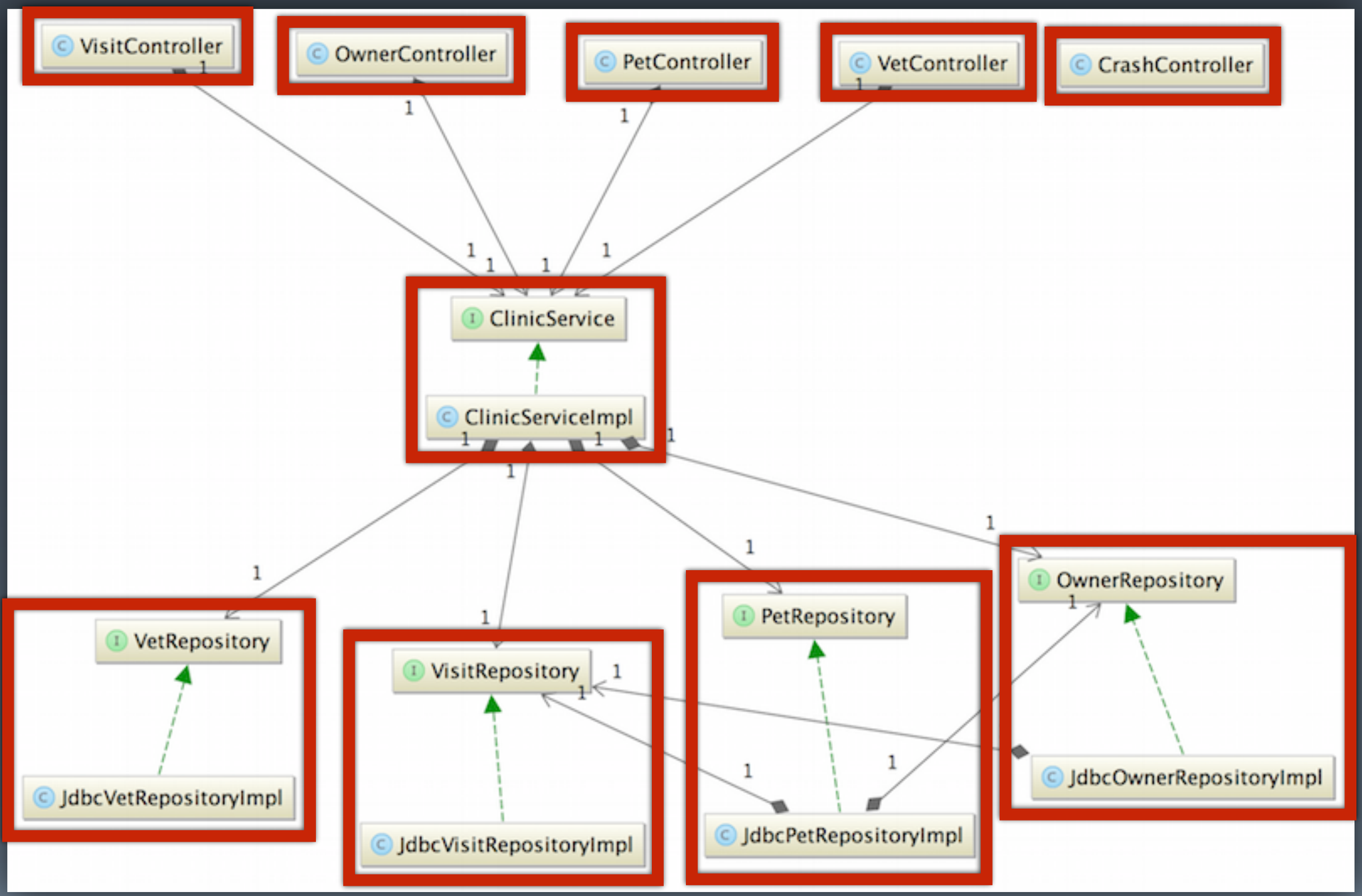




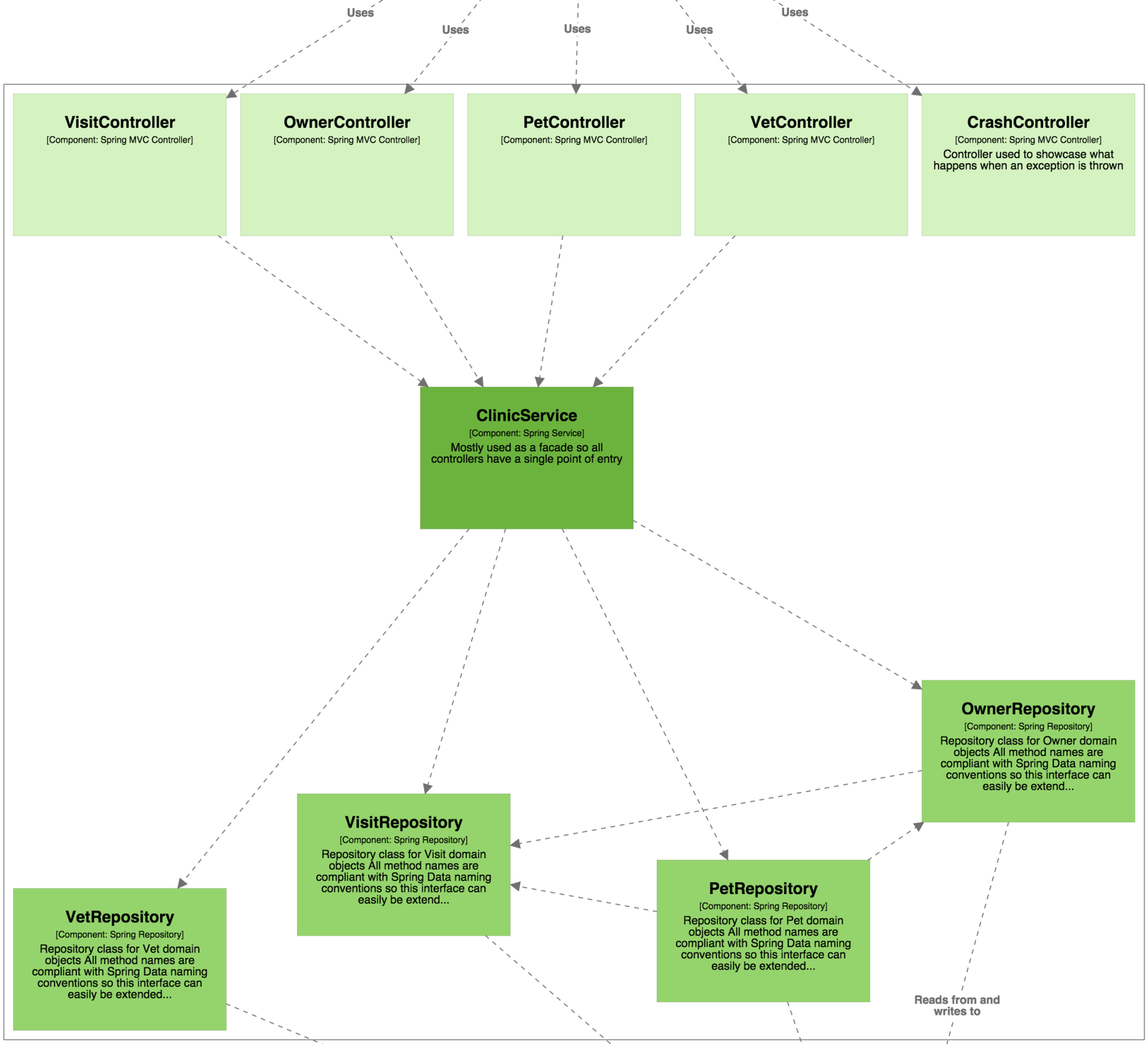
An auto-generated UML class diagram

What are the architecturally significant elements?





A UML class diagram showing architecturally significant elements



A component diagram, based upon the code

The code is the

embodiment

of the architecture

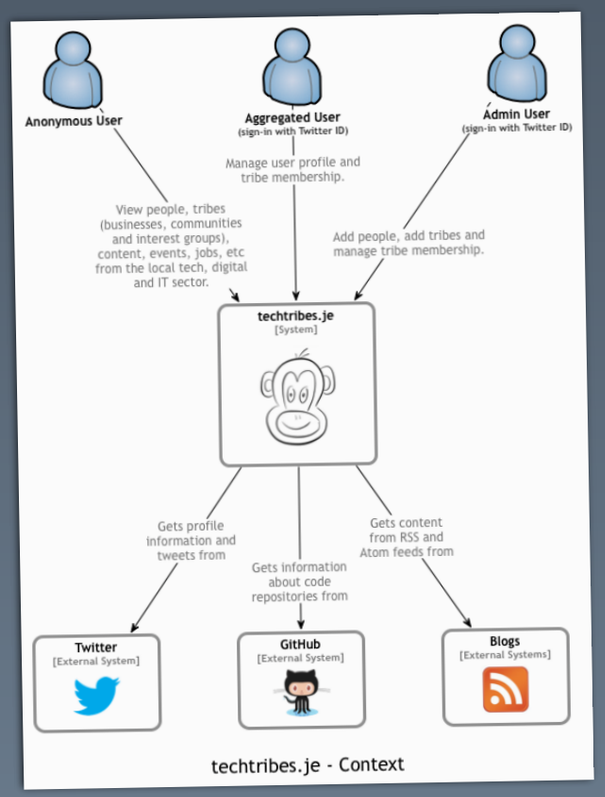
Is the architecture *in* the code?

In practice, architecture is embodied and ***recoverable from code****, and many languages provide architecture-level views of the system.*

A Survey of Architecture Description Languages

by Paul C. Clements

Context



People

Security groups/roles in configuration files, etc.

Software Systems

Integration points, APIs, known libraries, credentials for inbound consumers, etc.

Containers

IDE projects/modules, build output (code and infrastructure), etc.

Components

Extractable from the code if an architecturally-evident coding style has been adopted.

People

Security groups/roles in configuration files, etc.

Software Systems

Integration points, APIs, known libraries, credentials for inbound consumers, etc.

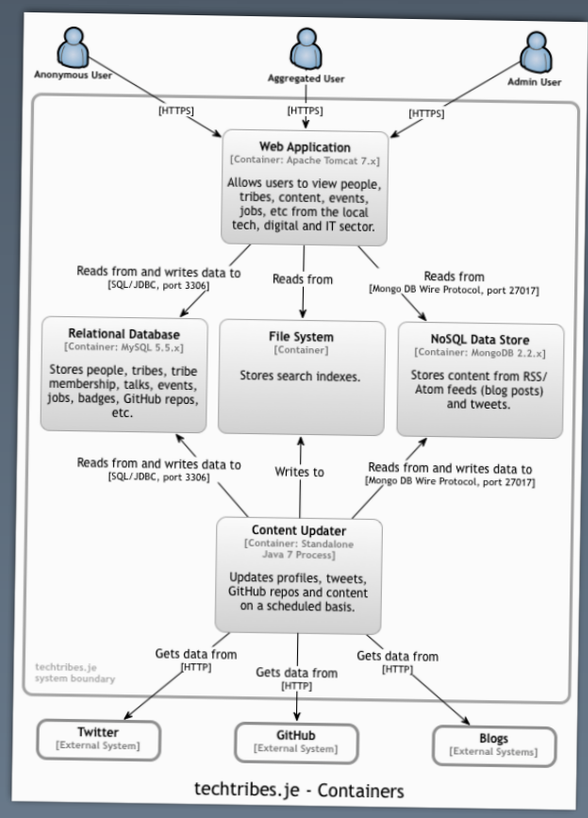
Containers

IDE projects/modules, build output (code and infrastructure), etc.

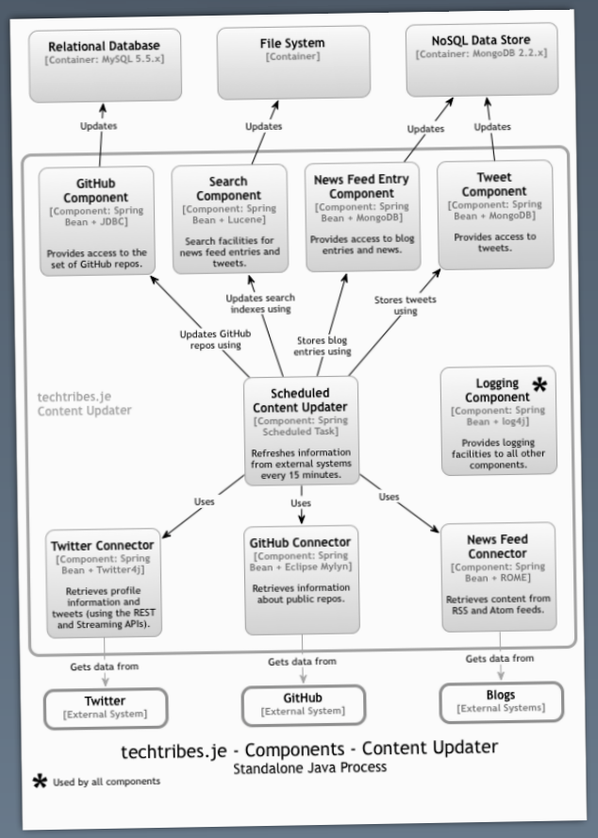
Components

Extractable from the code if an architecturally-evident coding style has been adopted.

Containers



Components



People

Security groups/roles in configuration files, etc.

Software Systems

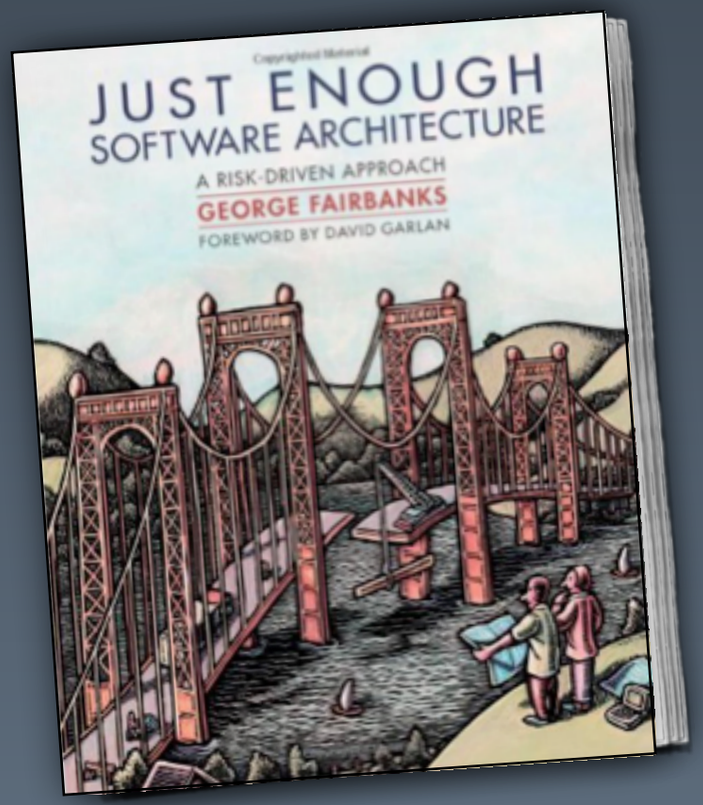
Integration points, APIs, known libraries, credentials for inbound consumers, etc.

Containers

IDE projects/modules, build output (code and infrastructure), etc.

Components

Extractable from the code if an architecturally-evident coding style has been adopted.



“architecturally-evident coding style”

Architecturally-evident coding styles include:

Annotations/attributes (`@Component`, `[Component]`, etc)

Naming conventions (`*Service`)

Namespacing/packaging

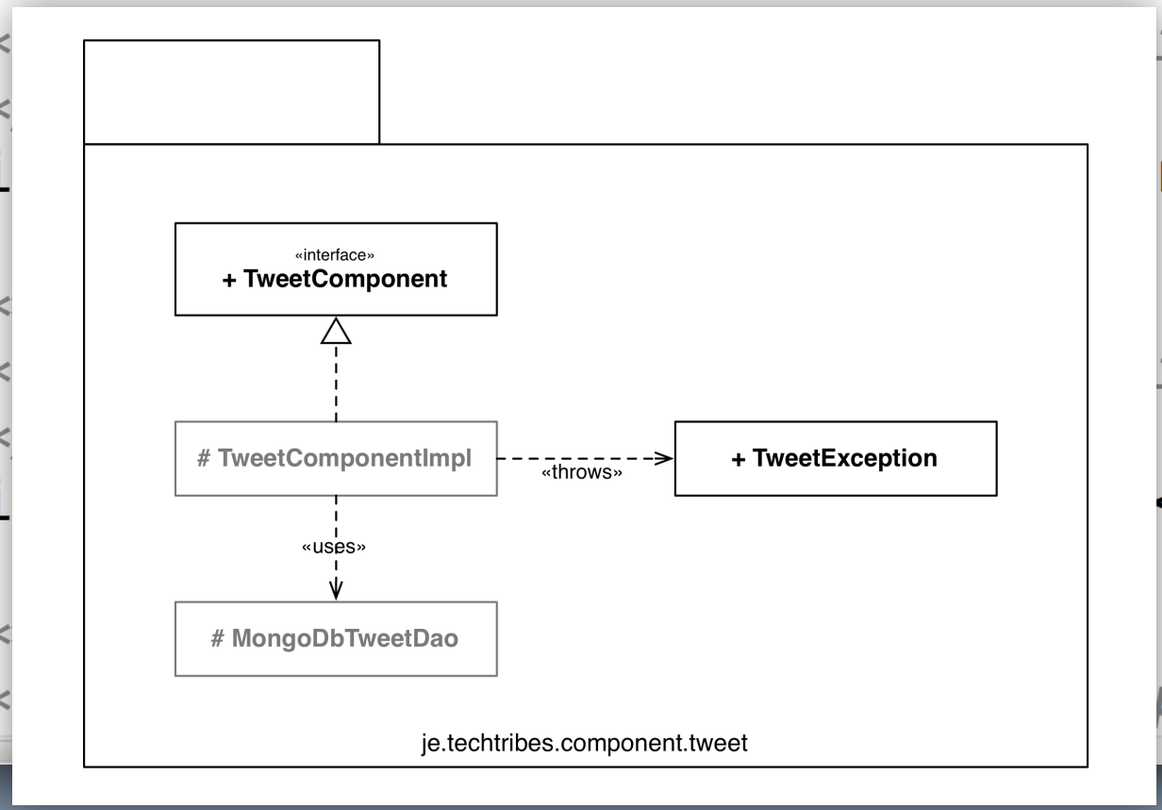
(`com.mycompany.system.components.*`)

Maven modules, OSGi modules, Java 9 and Jigsaw, JavaScript module patterns, ECMAScript 6 modules, microservices, etc

Project

techtribesje [techtribes] (~/sandbox/te
- docs
- lib
  - production
  - runtime
  - test
- structurizr
- techtribes-core
  - sql
  - src
    - je
      - techtribes
        - component
          - activity
          - badge
          - book
          - contentsource
          - creation
          - event
          - github
          - log
          - newsfeedentry
          - search
          - talk
          - tweet
            - component.xml
            - MongoDbTweetDao
            - TweetComponent
            - TweetComponentImpl
            - TweetException
          - domain
          - util
        - config.xml
  - test
  - techtribes-core.iml
- techtribes-updater
- techtribes-util
- techtribes-web
- .gitignore
- build.number

```
package je.techtribes.component.tweet;

import ...

/**
 * Provides access to tweets.
 */
@Component
public interface TweetComponent {

    /**
     * Gets the most recent tweets by page num
     */
    List<Tweet> getRecentTweets(int page, int

/**
 *
 *
Li

/*
 *
 *
Li

/*
 *
```



Modular Monoliths

Extract as much of the software architecture from the code as possible, and supplement where necessary

Create an architecture description language using code

```java
/**
 * This is a C4 representation of the Spring PetClinic sample app
 * (https://github.com/spring-projects/spring-petclinic/).
 *
 * Use the examples/springpetclinic.sh file to run this example -
 * you'll need a compiled version of the app on the CLASSPATH.
 */
public class SpringPetClinic {

    public static void main(String[] args) throws Exception {
        Workspace workspace = new Workspace("Spring PetClinic",
                "This is a C4 representation of the Spring PetClinic sample app (https://github.com/spring-pro
        Model model = workspace.getModel();

        // create the basic model (the stuff we can't get from the code)
        SoftwareSystem springPetClinic = model.addSoftwareSystem("Spring PetClinic", "Allows employees to view
        Person user = model.addPerson("Clinic Employee", "An employee of the clinic");
        user.uses(springPetClinic, "Uses");

        Container webApplication = springPetClinic.addContainer(
                "Web Application", "Allows employees to view and manage information regarding the veterinarians
        Container relationalDatabase = springPetClinic.addContainer(
                "Relational Database", "Stores information regarding the veterinarians, the clients, and their
        user.uses(webApplication, "Uses", "HTTP");
        webApplication.uses(relationalDatabase, "Reads from and writes to", "JDBC, port 9001");

        // and now automatically find all Spring @Controller, @Component, @Service and @Repository components
        ComponentFinder componentFinder = new ComponentFinder(
                webApplication, "org.springframework.samples.petclinic",
                new SpringComponentFinderStrategy(),
                new JavadocComponentFinderStrategy(new File("/Users/simon/Documents/sandbox/spring/spring-petc
        componentFinder.findComponents();

        // connect the user to all of the Spring MVC controllers
        webApplication.getComponents().stream()
                .filter(c -> c.getTechnology().equals("Spring MVC Controller"))
                .forEach(c -> user.uses(c, "Uses"));

        // connect all of the repository components to the relational database
```

```java
/**
 * This is a C4 representation of the Spring PetClinic sample app
 * (https://github.com/spring-projects/spring-petclinic/).
 *
 * Use the examples/springpetclinic.sh file to run this example -
 * you'll need a compiled version of the app on the CLASSPATH.
 */
public class SpringPetClinic {

    public static void main(String[] args) throws Exception {
        Workspace workspace = new Workspace("Spring PetClinic",
                "This is a C4 representation of the Spring PetClinic sample app (https://github.com/spring-pro
        Model model = workspace.getModel();

        // create the basic model (the stuff we can't get from the code)
        SoftwareSystem springPetClinic = model.addSoftwareSystem("Spring PetClinic", "Allows employees to view
        Person user = model.addPerson("Clinic Employee", "An employee of the clinic");
        user.uses(springPetClinic, "Uses");

        Container webApplication = springPetClinic.addContainer(
                "Web Application", "Allows employees to view and manage information regarding the veterinarians
        Container relationalDatabase = springPetClinic.addContainer(
                "Relational Database", "Stores information regarding the veterinarians, the clients, and their
        user.uses(webApplication, "Uses", "HTTP");
        webApplication.uses(relationalDatabase, "Reads from and writes to", "JDBC, port 9001");

        // and now automatically find all Spring @Controller, @Component, @Service and @Repository components
        ComponentFinder componentFinder = new ComponentFinder(
                webApplication, "org.springframework.samples.petclinic",
                new SpringComponentFinderStrategy(),
                new JavadocComponentFinderStrategy(new File("/Users/simon/Documents/sandbox/spring/spring-petc
        componentFinder.findComponents();

        // connect the user to all of the Spring MVC controllers
        webApplication.getComponents().stream()
                .filter(c -> c.getTechnology().equals("Spring MVC Controller"))
                .forEach(c -> user.uses(c, "Uses"));

        // connect all of the repository components to the relational database
```

```
// and now automatically find all Spring @Controller, @Component, @Service and @Repository components
ComponentFinder componentFinder = new ComponentFinder(
        webApplication, "org.springframework.samples.petclinic",
        new SpringComponentFinderStrategy(),
        new JavadocComponentFinderStrategy(new File("/Users/simon/Documents/sandbox/spring/spring-petc
componentFinder.findComponents();

// connect the user to all of the Spring MVC controllers
webApplication.getComponents().stream()
        .filter(c -> c.getTechnology().equals("Spring MVC Controller"))
        .forEach(c -> user.uses(c, "Uses"));

// connect all of the repository components to the relational database
webApplication.getComponents().stream()
        .filter(c -> c.getTechnology().equals("Spring Repository"))
        .forEach(c -> c.uses(relationalDatabase, "Reads from and writes to"));

// finally create some views
ViewSet viewSet = workspace.getViews();
SystemContextView contextView = viewSet.createContextView(springPetClinic);
contextView.addAllSoftwareSystems();
contextView.addAllPeople();

ContainerView containerView = viewSet.createContainerView(springPetClinic);
containerView.addAllPeople();
containerView.addAllSoftwareSystems();
containerView.addAllContainers();

ComponentView componentView = viewSet.createComponentView(webApplication);
componentView.addAllComponents();
componentView.addAllPeople();
componentView.add(relationalDatabase);

// link the architecture model with the code
for (Component component : webApplication.getComponents()) {
    if (component.getSourcePath() != null) {
        component.setSourcePath(component.getSourcePath().replace(
                "/Users/simon/Documents/sandbox/spring/spring-petclinic/",
```

```
// finally create some views
ViewSet viewSet = workspace.getViews();
SystemContextView contextView = viewSet.createContextView(springPetClinic);
contextView.addAllSoftwareSystems();
contextView.addAllPeople();

ContainerView containerView = viewSet.createContainerView(springPetClinic);
containerView.addAllPeople();
containerView.addAllSoftwareSystems();
containerView.addAllContainers();

ComponentView componentView = viewSet.createComponentView(webApplication);
componentView.addAllComponents();
componentView.addAllPeople();
componentView.add(relationalDatabase);
```

Structurizr for Java

(open source on GitHub)

GitHub This repository Search Explore Features Enterprise Blog Sign up Sign in

structurizr / java Watch 24 Star 59 Fork 30

Java tools

131 commits 1 branch 0 releases 3 contributors

branch: master java / +

Added a software architecture model for the Java EE Hands on Lab "Mov…

simonbrowndotje authored 14 days ago latest commit 90dea447f9

gradle	Gradle Support: cleanup workspace, commit gradle wrapper, remove ant/ivy	2 months ago
structurizr-annotations/src/co…	Some refactoring and an initial version of a Javadoc component finder…	25 days ago
structurizr-client	This saves a little bandwidth. :-)	14 days ago
structurizr-core	Fixed a bug where layout information wasn't getting copied when inter…	14 days ago
structurizr-examples	Added a software architecture model for the Java EE Hands on Lab "Mov…	14 days ago
structurizr-spring	Added a sourcePath property to components ... which is automatically …	23 days ago
.gitignore	Added some comments.	2 months ago
LICENSE	Initial commit	10 months ago
README.md	Added build instructions.	2 months ago
build.gradle	Changed styles -> configuration.styles	27 days ago
copyJars.sh	Added fixed some bugs and added an initial implementation to copy lay…	2 months ago
gradle.properties	Renamed components and messing with Maven repo publication.	2 months ago
gradlew	Trimmed down the Gradle config and stopped tracking the IDEA project …	2 months ago
gradlew.bat	Gradle support	2 months ago
settings.gradle	Renamed components and messing with Maven repo publication.	2 months ago

Code Issues 1 Pull requests 1 Pulse Graphs

HTTPS clone URL https://github.com

You can clone with HTTPS or Subversion.

Clone in Desktop Download ZIP

README.md

Structurizr for Java

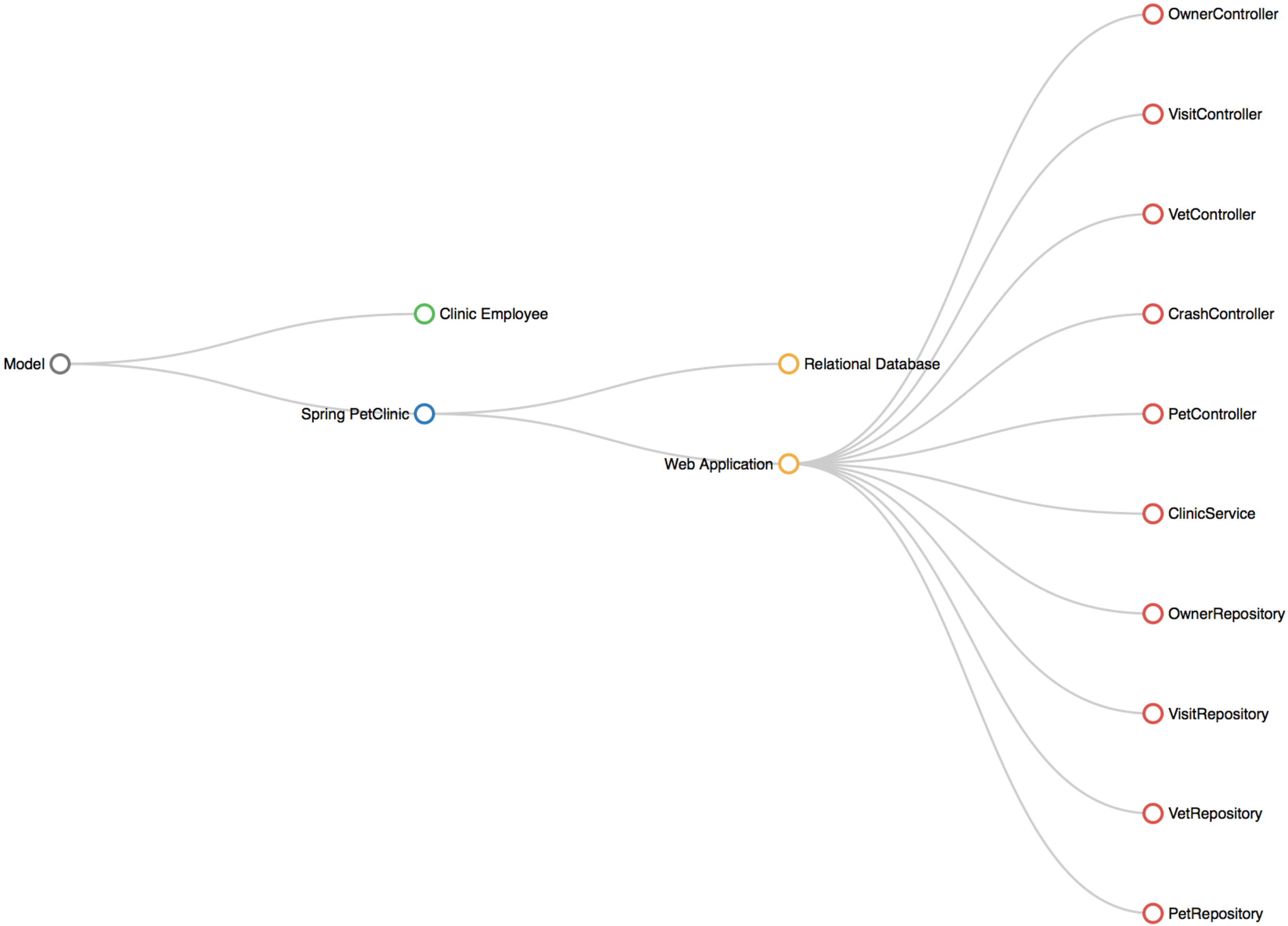
Structurizr is an implementation of the C4 model as described in Simon Brown's Software Architecture for Developers book, which provides a way to easily and effectively communicate the software architecture of a software system. Structurizr allows you to create **software architecture models and diagrams as code**. This project contains the Java implementation and tooling.

Everything you see here is a work in progress. See www.structurizr.com for more information.

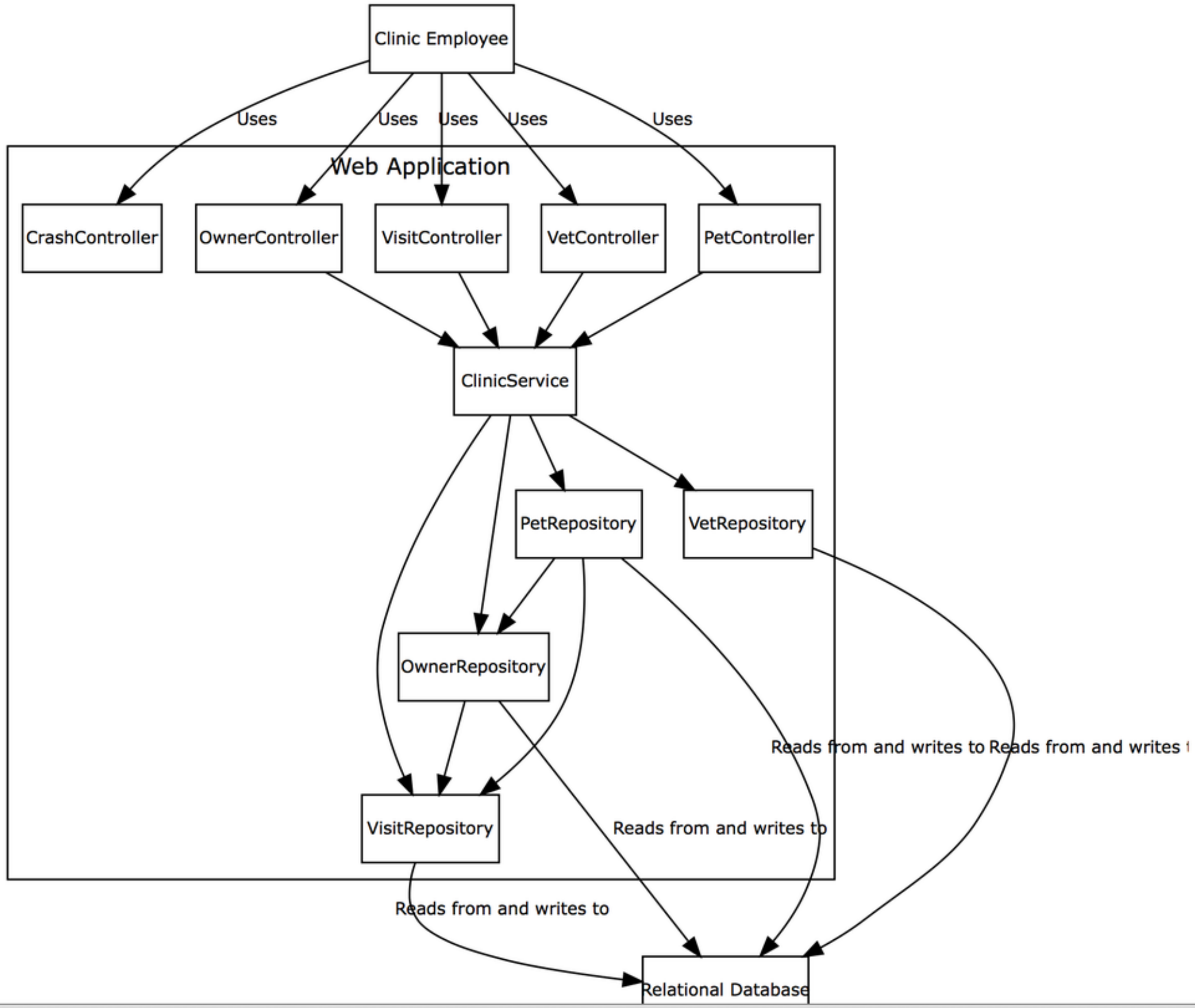
Building

To build Struturizr for Java from the sources (you'll need Java 8)...

```
git clone https://github.com/structurizr/java.git

./gradlew build
```


Model
Clinic Employee
Spring PetClinic
Relational Database
Web Application
OwnerController
VisitController
VetController
CrashController
PetController
ClinicService
OwnerRepository
VisitRepository
VetRepository
PetRepository

Spring PetClinic - Web Application - Components



Query 1 — this query has been used to initialize the console

Test

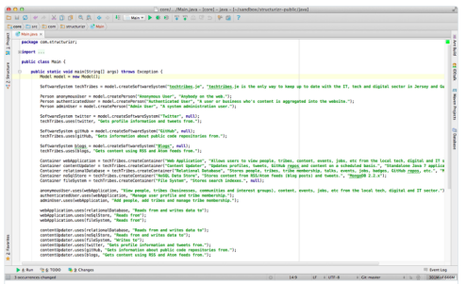
Neo4j GraphGist | Resources | Page Source | GitHub Gist/File / Dropbox URL

Architecture as a Graph (AaaG)

Techtribes.je technical architecture as a graph model

No Errors. Examples! G+1 Like 0 Tweet 0 Rate This

Simon Brown, Robin Bramley and Michael Hunger had a discussion on twitter about architecture modeling using code/a DSL, started by:



Simon Brown @simonbrown — Follow

I've been messing with this stuff over the past few days … architecture model as code?

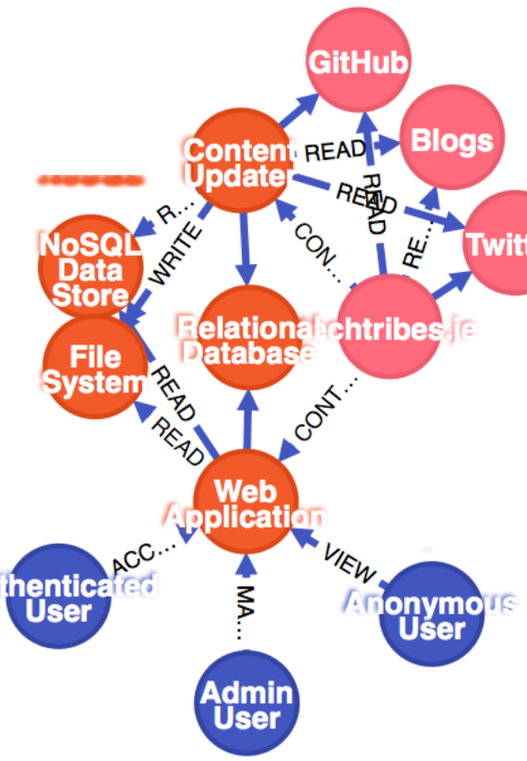
8:57 AM - 21 Jun 2014

3 5

Techtribes.je is the only way to keep up to date with the IT, tech and digital sector in Jersey and Guernsey, Channel Islands.

This interactive graph document describes its system architecture, and demonstrates some use-cases.

Setup



What can the different users do with which software

Query 2

```
MATCH (u:User)-[r]->(s:Software)
RETURN u.name, type(r), r.description, s.name
```

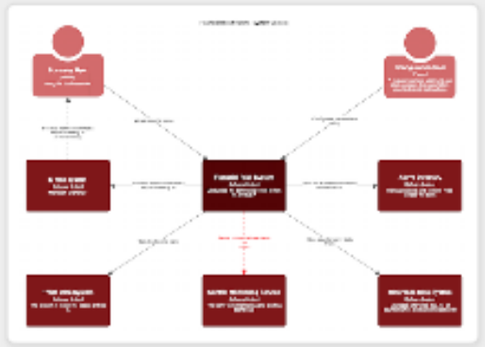
Test

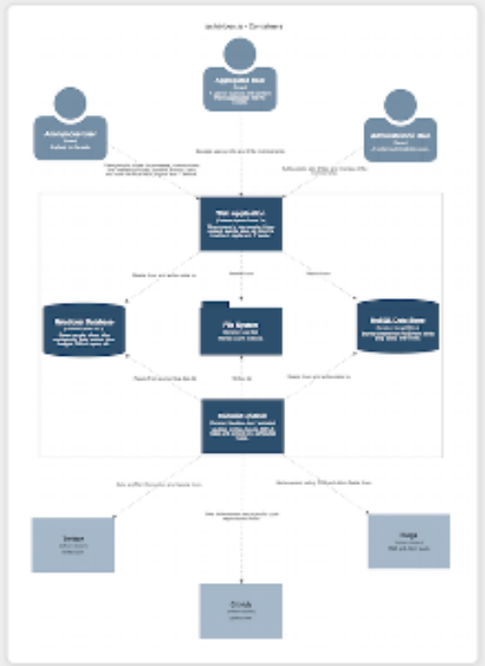
u.name	type(r)	r.description	s.name

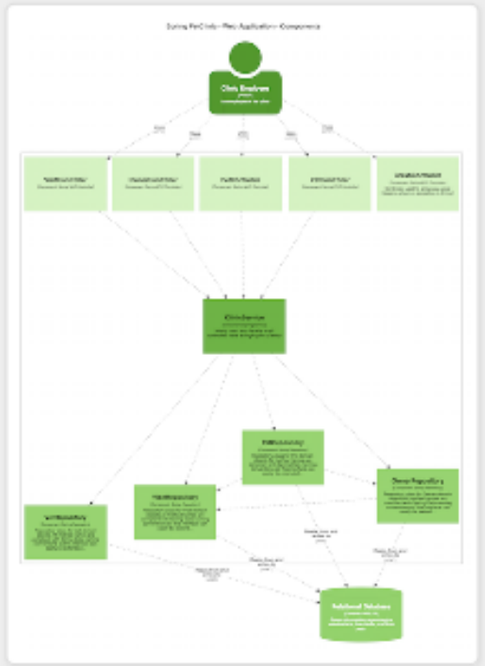
Vendor alert!

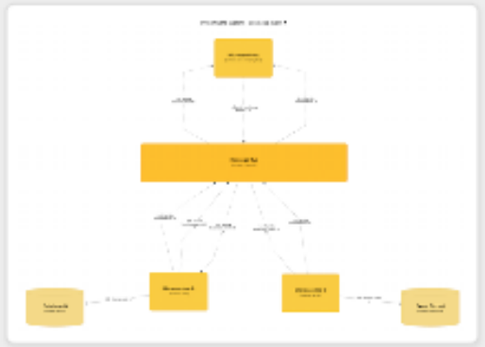
Structurizr

Create web-based software architecture diagrams using code.







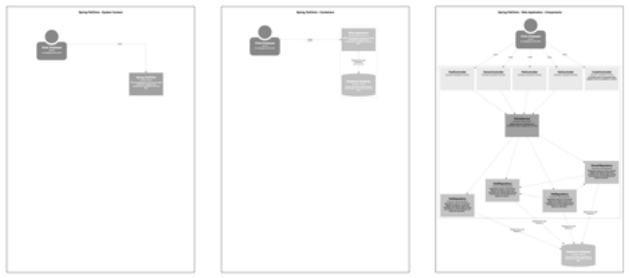


What is Structurizr?

Structurizr provides a way to create software architecture models as code, which can then be visualised in your web browser. Structurizr is *not* a conventional general purpose diagramming tool like Microsoft Visio. In a nutshell, you write some code to describe the software architecture of your software, upload the resulting model (as a JSON document) via the web API

Structurizr Client

2. Upload the software architecture model to Structurizr using the web API



1. Write one or more programs to create a software architecture model of your software system

Manually create model elements using code or extract them from your software system by parsing source code or using reflection and static analysis techniques

Software System

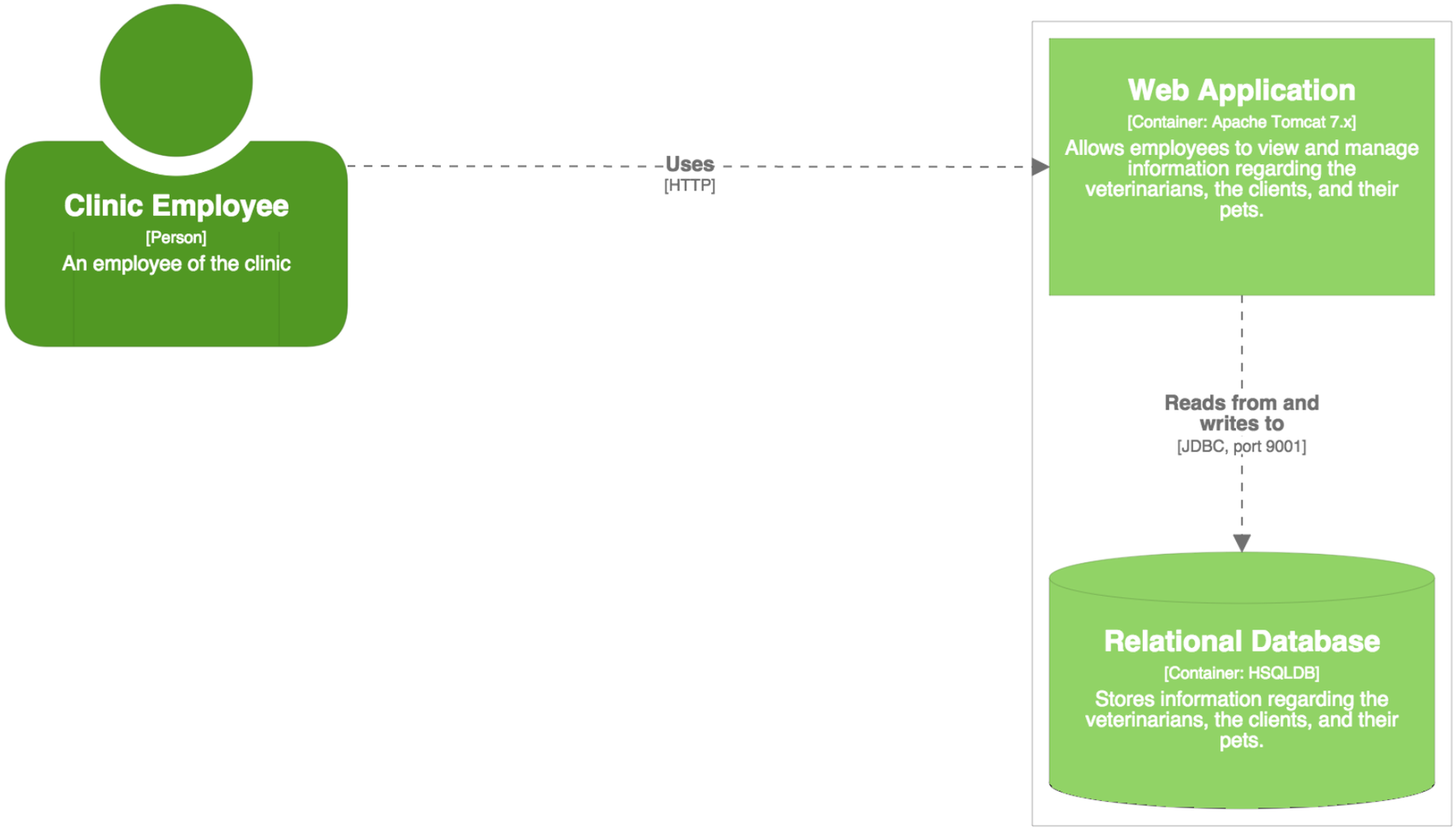
3. Explore, visualise and share your software architecture model with web-based software architecture diagrams

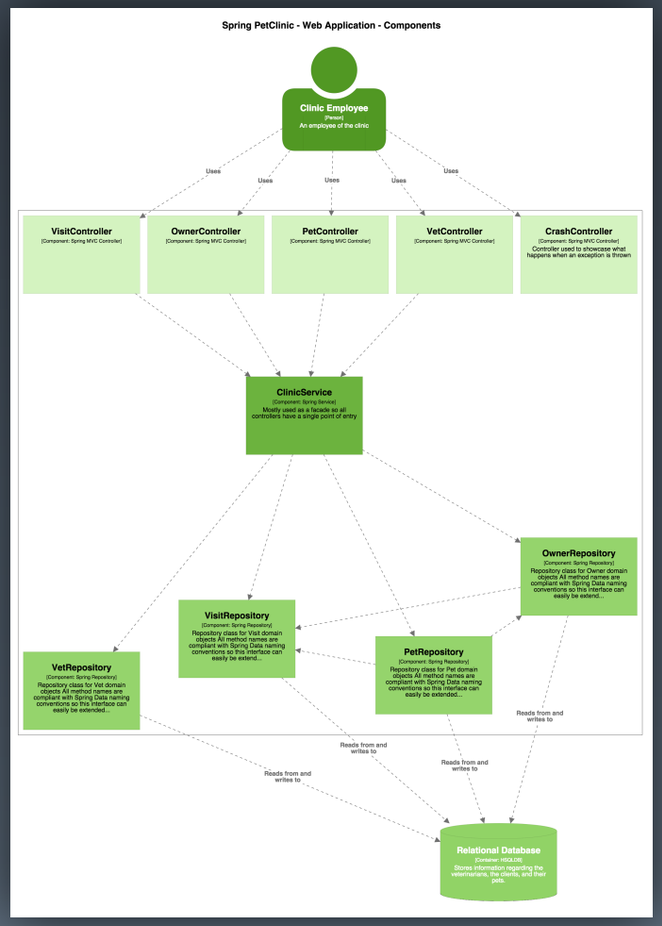
Structurizr is a web-based tool that creates diagrams for you, based upon the software architecture model and views defined in your code

What is Structurizr?

Spring PetClinic - System Context

Spring PetClinic - Containers

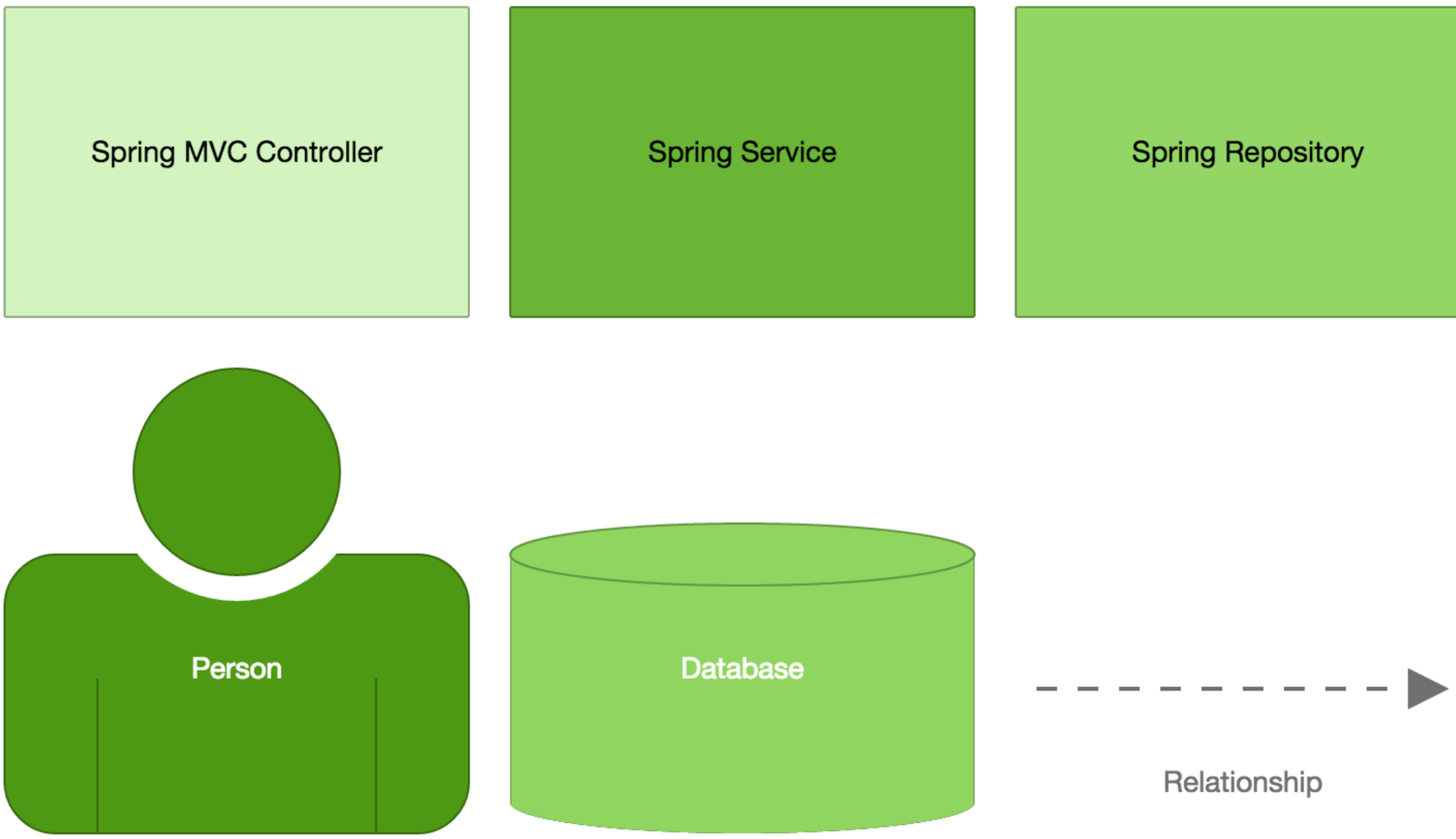




Spring PetClinic - Web Application - Components

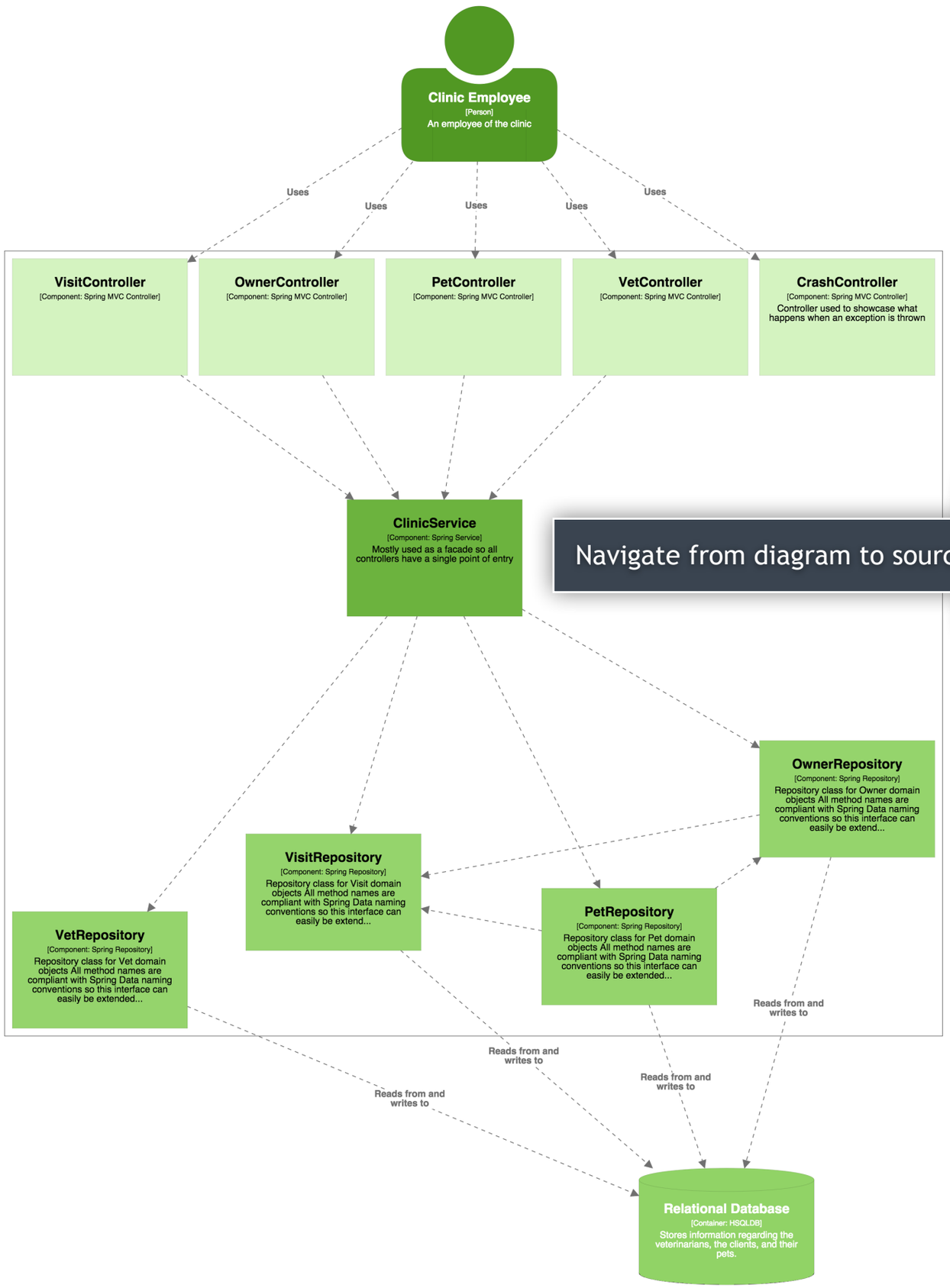
Clinic Employee
[Person]
An employee of the clinic
Uses
VisitController
[Component: Spring MVC Controller]
OwnerController
[Component: Spring MVC Controller]
PetController
[Component: Spring MVC Controller]
VetController
[Component: Spring MVC Controller]
CrashController
[Component: Spring MVC Controller]
Controller used to showcase what happens when an exception is thrown
ClinicService
[Component: Spring Service]
Mostly used as a facade so all controllers have a single point of entry
OwnerRepository
[Component: Spring Repository]
Repository class for Owner domain objects All method names are compliant with Spring Data naming conventions so this interface can easily be extend...
VisitRepository
[Component: Spring Repository]
Repository class for Visit domain objects All method names are compliant with Spring Data naming conventions so this interface can easily be extend...
PetRepository
[Component: Spring Repository]
Repository class for Pet domain objects All method names are compliant with Spring Data naming conventions so this interface can easily be extended...
VetRepository
[Component: Spring Repository]
Repository class for Vet domain objects All method names are compliant with Spring Data naming conventions so this interface can easily be extended...
Reads from and writes to

Spring PetClinic - Web Application - Components



Close

Spring PetClinic - Web Application - Components



Navigate from diagram to source code

GitHub This repository Search Explore Features Enterprise Pricing Sign up Sign in

spring-projects / spring-petclinic Watch 174 Star 614 Fork 1,070

Branch: master

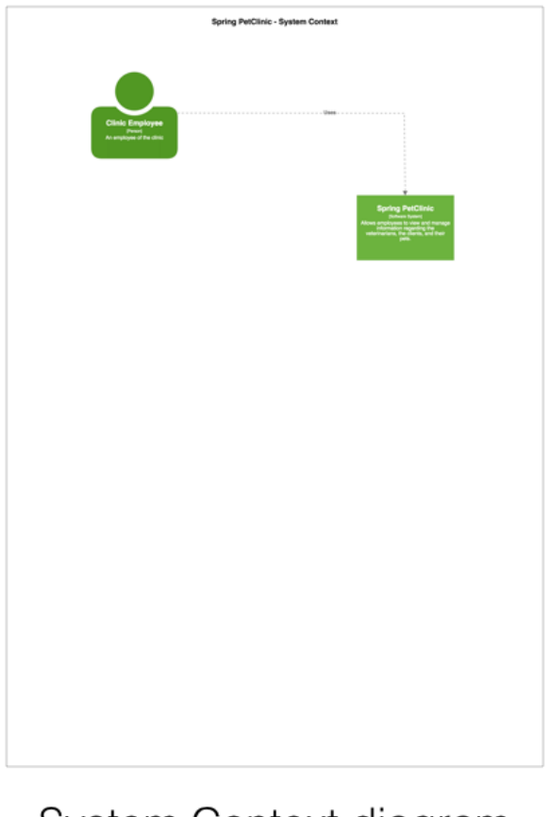
spring-petclinic / src / main / java / org / springframework / samples / petclinic / service / **ClinicService.java**

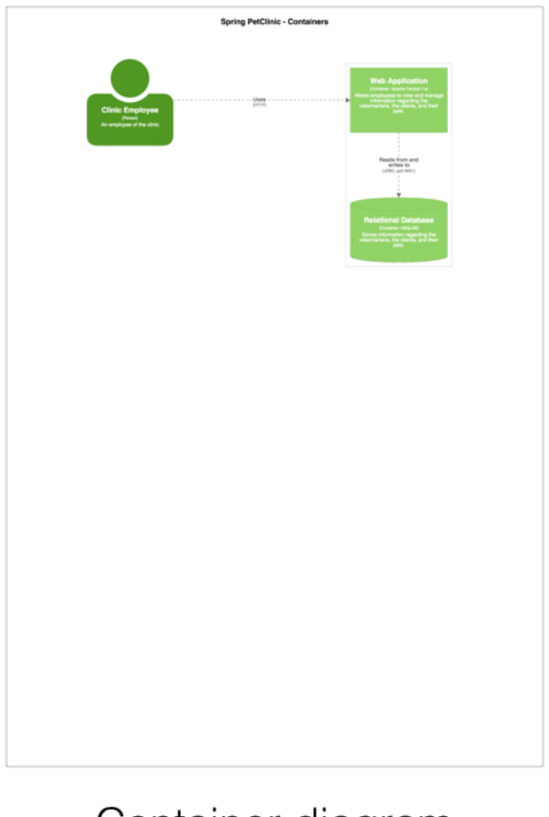
michaelisvy test methods:used should/shouldNot 5c9ab6 on 16 Jan

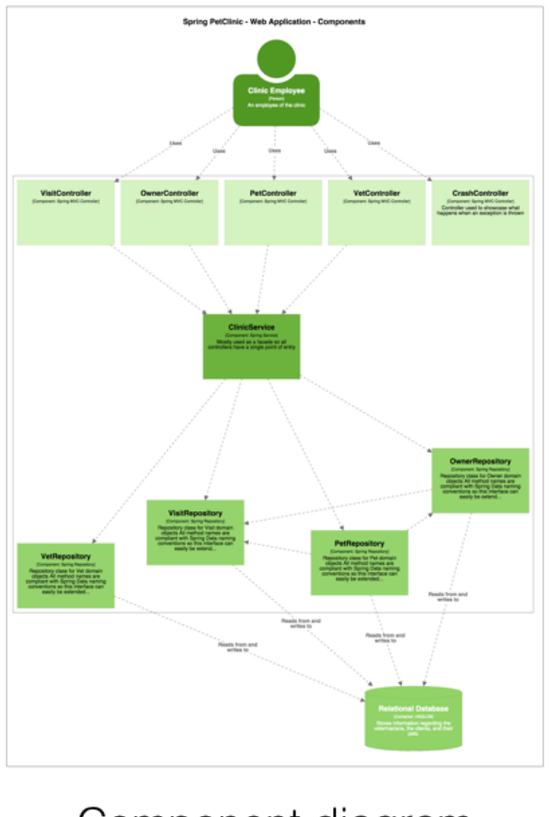
2 contributors

52 lines (38 sloc) 1.67 KB Raw Blame History

```
1  /*
2   * Copyright 2002-2013 the original author or authors.
3   *
4   * Licensed under the Apache License, Version 2.0 (the "License");
5   * you may not use this file except in compliance with the License.
6   * You may obtain a copy of the License at
7   *
8   *      http://www.apache.org/licenses/LICENSE-2.0
9   *
10  * Unless required by applicable law or agreed to in writing, software
11  * distributed under the License is distributed on an "AS IS" BASIS,
12  * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13  * See the License for the specific language governing permissions and
14  * limitations under the License.
15  */
16 package org.springframework.samples.petclinic.service;
17
18 import java.util.Collection;
19
20 import org.springframework.dao.DataAccessException;
21 import org.springframework.samples.petclinic.model.Owner;
22 import org.springframework.samples.petclinic.model.Pet;
23 import org.springframework.samples.petclinic.model.PetType;
24 import org.springframework.samples.petclinic.model.Vet;
25 import org.springframework.samples.petclinic.model.Visit;
26
27
28 /**
29  * Mostly used as a facade so all controllers have a single point of entry
30  *
31  * @author Michael Isvy
32  */
33 public interface ClinicService {
34
35     Collection<PetType> findPetTypes() throws DataAccessException;
36
37     Owner findOwnerById(int id) throws DataAccessException;
38
39     Pet findPetById(int id) throws DataAccessException;
40
41     void savePet(Pet pet) throws DataAccessException;
42
43     void saveVisit(Visit visit) throws DataAccessException;
44
45     Collection<Vet> findVets() throws DataAccessException;
46
47     void saveOwner(Owner owner) throws DataAccessException;
48
49     Collection<Owner> findOwnerByLastName(String lastName) throws DataAccessException;
50
51 }
```







System Context diagram | Container diagram | Component diagram | Source code

Double-click a software system

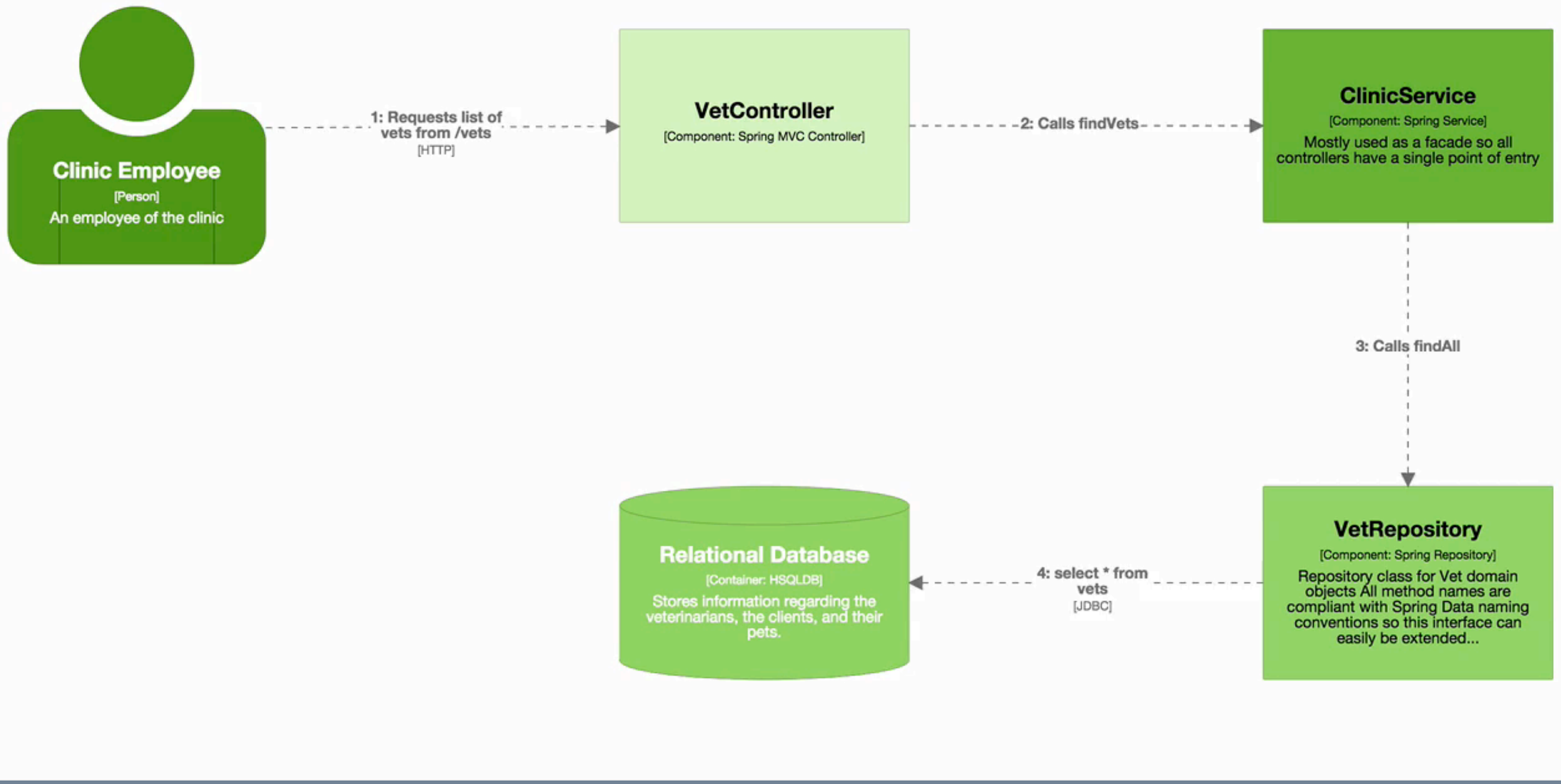
Double-click a container

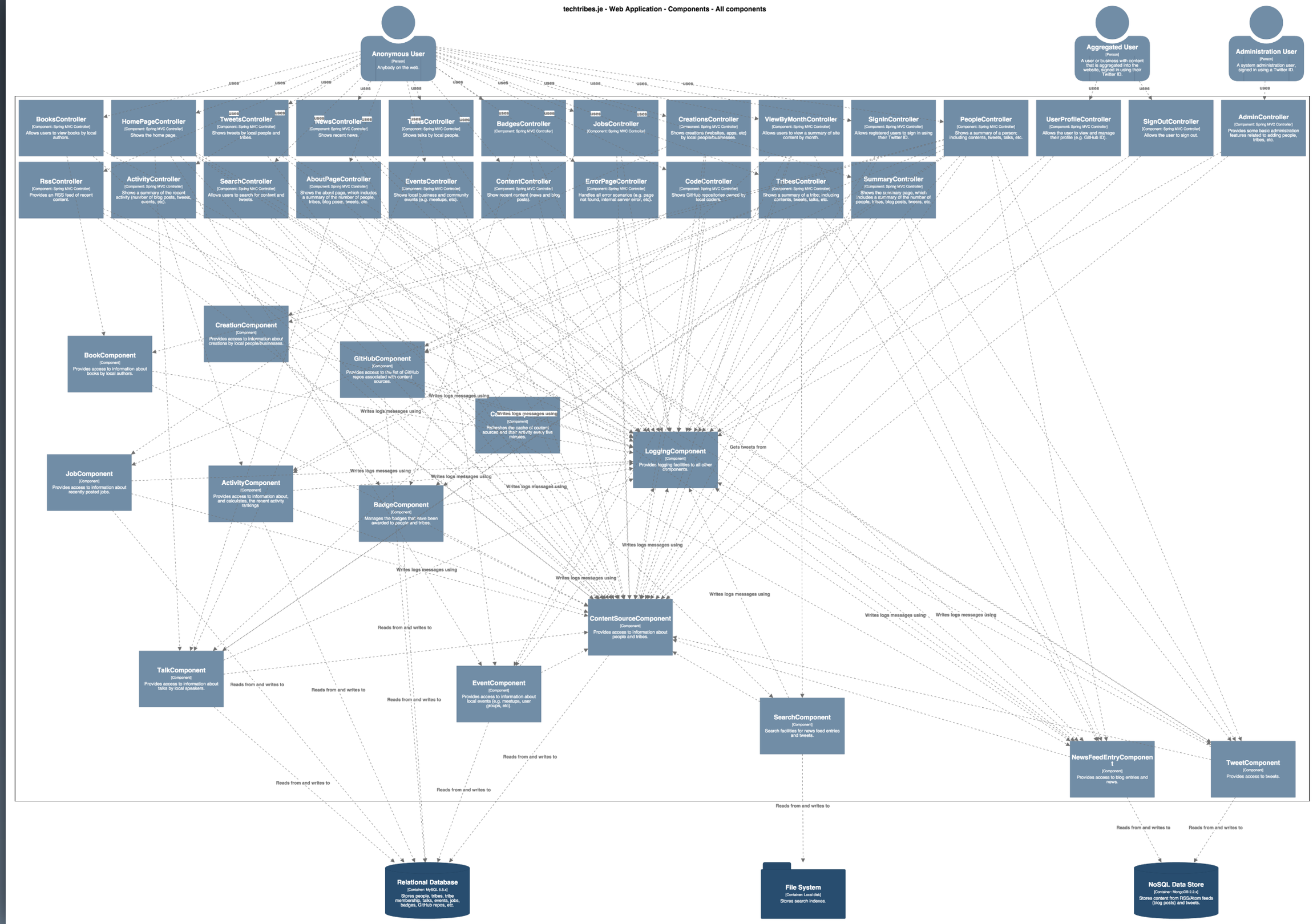
Double-click a component

Diagrams are maps

```
Component vetRepository = webApplication.getComponentWithName("VetRepository");

DynamicView dynamicView = viewSet.createDynamicView(springPetClinic, "View list of vets");
dynamicView.add(clinicEmployee, "Requests list of vets from /vets", vetController);
dynamicView.add(vetController, "Calls findVets", clinicService);
dynamicView.add(clinicService, "Calls findAll", vetRepository);
dynamicView.add(vetRepository, "select * from vets", relationalDatabase);
```

Spring PetClinic - View list of vets

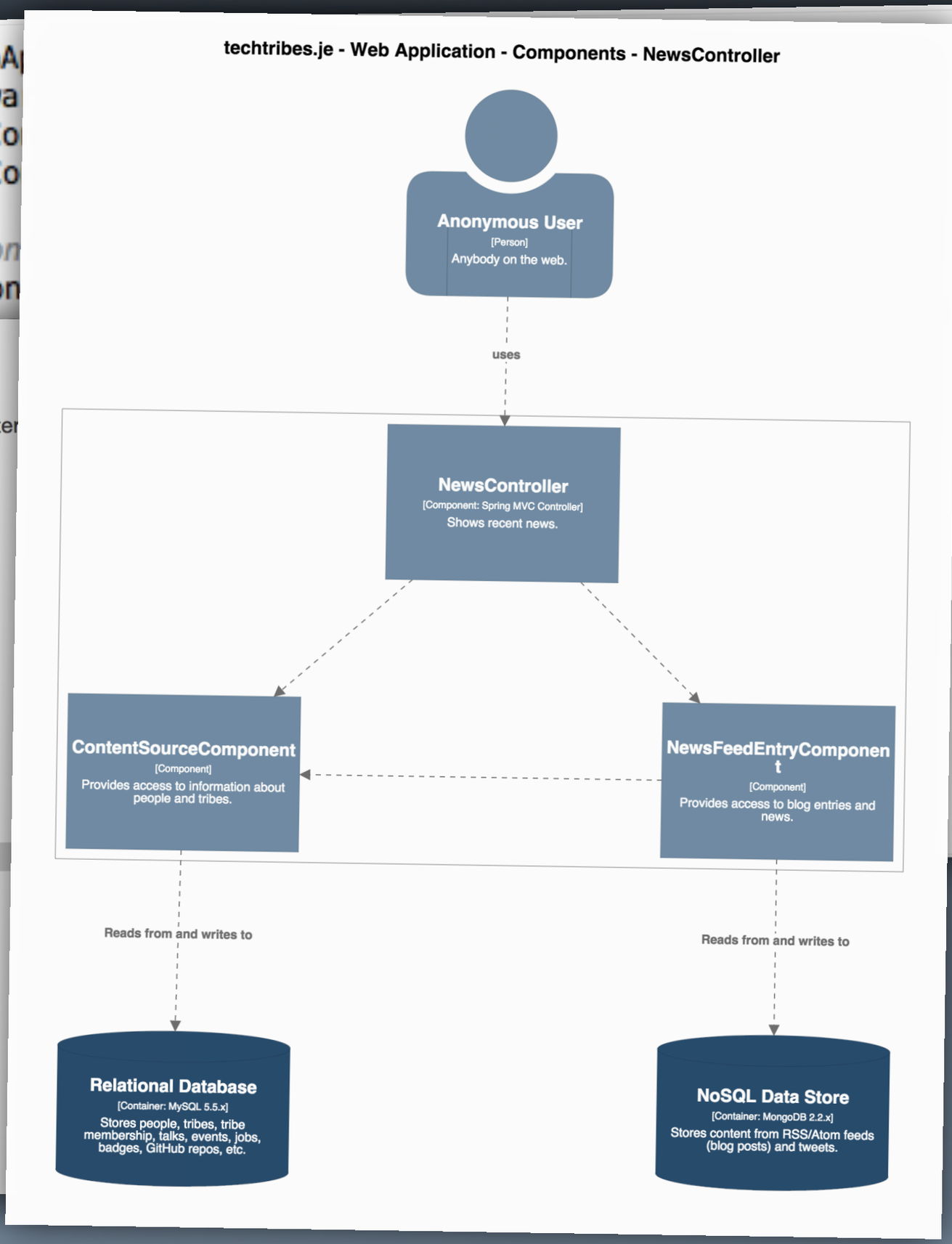




Does this approach scale?

```
private static void createComponentViewsForWebA
    SoftwareSystem techTribes = model.getSoftwa
    Container contentUpdater = techTribes.getCo
    Container webApplication = techTribes.getCo

    // create one component view per Spring con
    Set<Component> controllers = webApplication
```

techtribes.je - System Context
techtribes.je - Containers
techtribes.je - Content Updater - Components - Awarding badges
techtribes.je - Content Updater - Components - Updating information from exter
techtribes.je - Content Updater - Components - Updating recent activity
techtribes.je - Web Application - Components - AboutPageController
techtribes.je - Web Application - Components - ActivityController
techtribes.je - Web Application - Components - AdminController
✓ techtribes.je - Web Application - Components - All components
techtribes.je - Web Application - Components - BadgesController
techtribes.je - Web Application - Components - BooksController
techtribes.je - Web Application - Components - CodeController
techtribes.je - Web Application - Components - ContentController
techtribes.je - Web Application - Components - CreationsController
techtribes.je - Web Application - Components - ErrorPageController
techtribes.je - Web Application - Components - EventsController
techtribes.je - Web Application - Components - HomePageController
techtribes.je - Web Application - Components - JobsController
techtribes.je - Web Application - Components - NewsController
techtribes.je - Web Application - Components - PeopleController
techtribes.je - Web Application - Components - RssController
techtribes.je - Web Application - Components - SearchController
techtribes.je - Web Application - Components - SignInController
techtribes.je - Web Application - Components - SignOutController
techtribes.je - Web Application - Components - SummaryController
techtribes.je - Web Application - Components - TalksController
techtribes.je - Web Application - Components - TribesController
techtribes.je - Web Application - Components - TweetsController
techtribes.je - Web Application - Components - UserProfileController
techtribes.je - Web Application - Components - ViewByMonthController



A model as code provides opportunities…

TeamCity

<APACHE ANT>

gradle

maven

Jenkins

Build pipeline integration keeps software architecture models up-to-date

If the software architecture model is *in* the code, it can be extracted *from* the code

simon.brown@codingthearchitecture.com
@simonbrown on Twitter