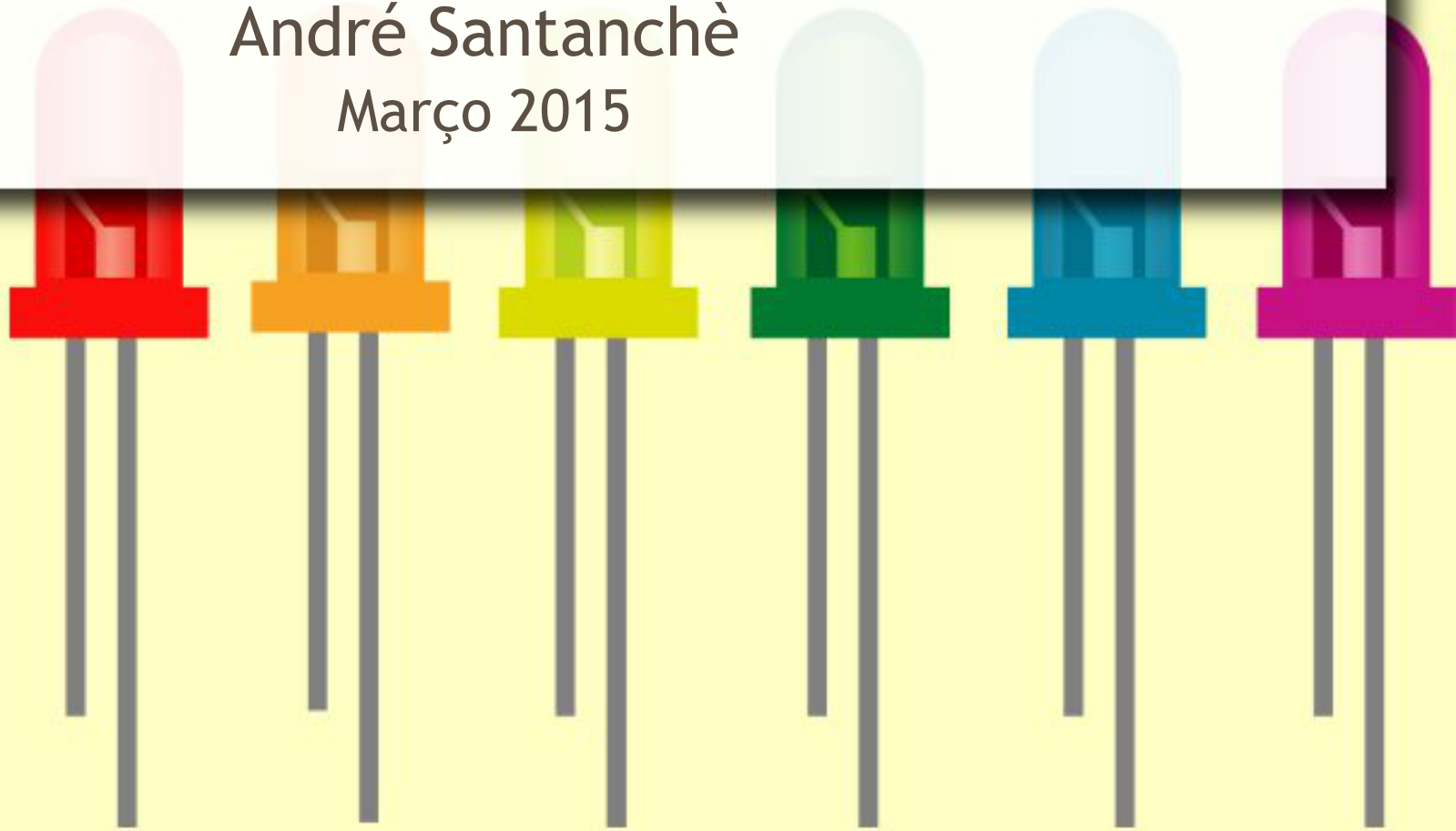
Programação Orientada a Objetos

Componentes de Software

André Santanchè

Março 2015



Componentes

- “Aquilo que entra na composição de alguma coisa.” (Aurélio, 2004)

- “que ou o que compõe ou ajuda na composição de algo” (Houaiss, 2006)

Porque usar componentes?

- Componentes na engenharia:
  "Sem dúvida nós produzimos software usando técnicas ultrapassadas. Sem dúvida nós ficamos com o lado curto do palitinho em confrontos com as pessoas de hardware porque eles são os industriais e nós somos os lavradores." (Mcilroy, 1968)

Tradução do original feita pelo autor: "We undoubtedly produce software by backward techniques. We undoubtedly get the short end of the stick in confrontations with hardware people because they are the industrialists and we are the crofters." (Mcilroy, 1968)

Composição

- “Composition enables prefabricated 'things' to be reused by rearranging them in ever-new composities”. (Szyperski, 2002)

O que é um componente?

- “Today, few terms in the software industry are less precise than component software.” (Olsen, 2006)

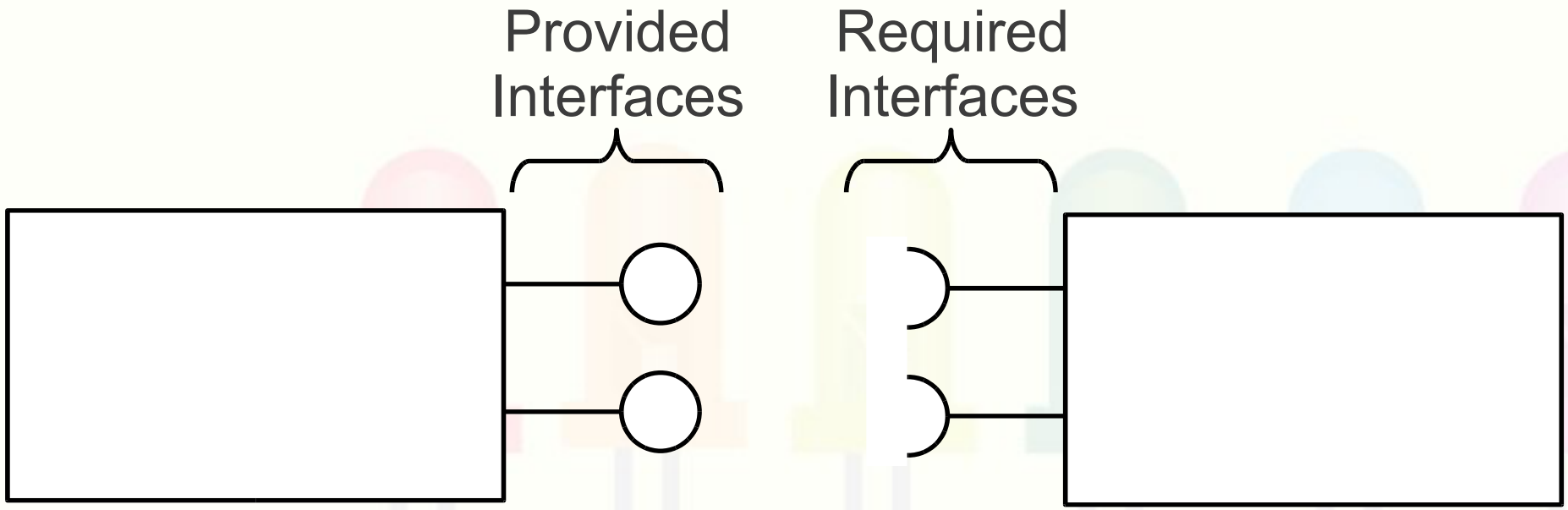
O que é um componente?
Características Comuns

- Entidade concebida para ser composta
  - do latim *componens*, derivado de *componere*, que quer dizer “colocar junto”.
- Publica sua funcionalidade através de uma interface
  - interface guia relacionamento componente x ambiente
- Componentes podem ser aninhados em outros componentes
  - componentes e sub-componentes

O que é um componente?
Características Desejáveis

- Contém código binário que implementa a funcionalidade declarada na interface
- Serviços acessíveis exclusivamente pela interface (black-box)
- Pacote padrão para distribuição

Diagrama de Componentes



Componentes x Objetos

- Componentes são unidades de distribuição, objetos não. (Szyperski, 2002)

Estudo de Caso

Componentização Sucessiva

Estudo de Caso

Componentização Sucessiva

- Programa para gerar identificadores únicos sequenciais.
- Mostra uma componentização em granularidades crescentes.
- Compara o papel de classes e componentes.

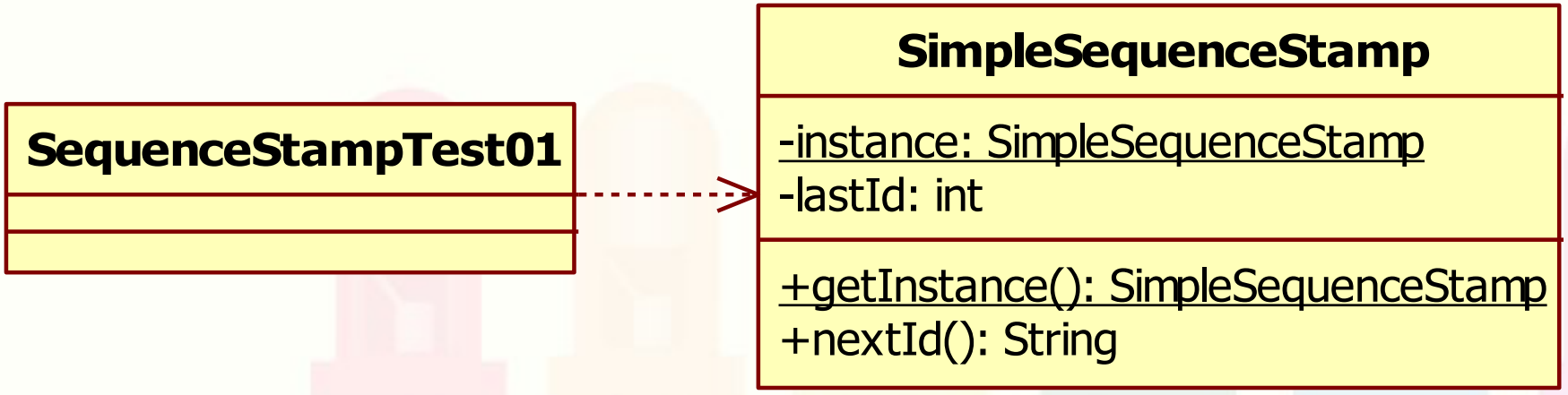
Componentização Sucessiva
Primeira Versão

Gerar identificador simples

Uso do Singleton

Primeira Versão

Uso do Padrão Singleton



Componentização Sucessiva
Segunda Versão

Gerar identificador simples e URI

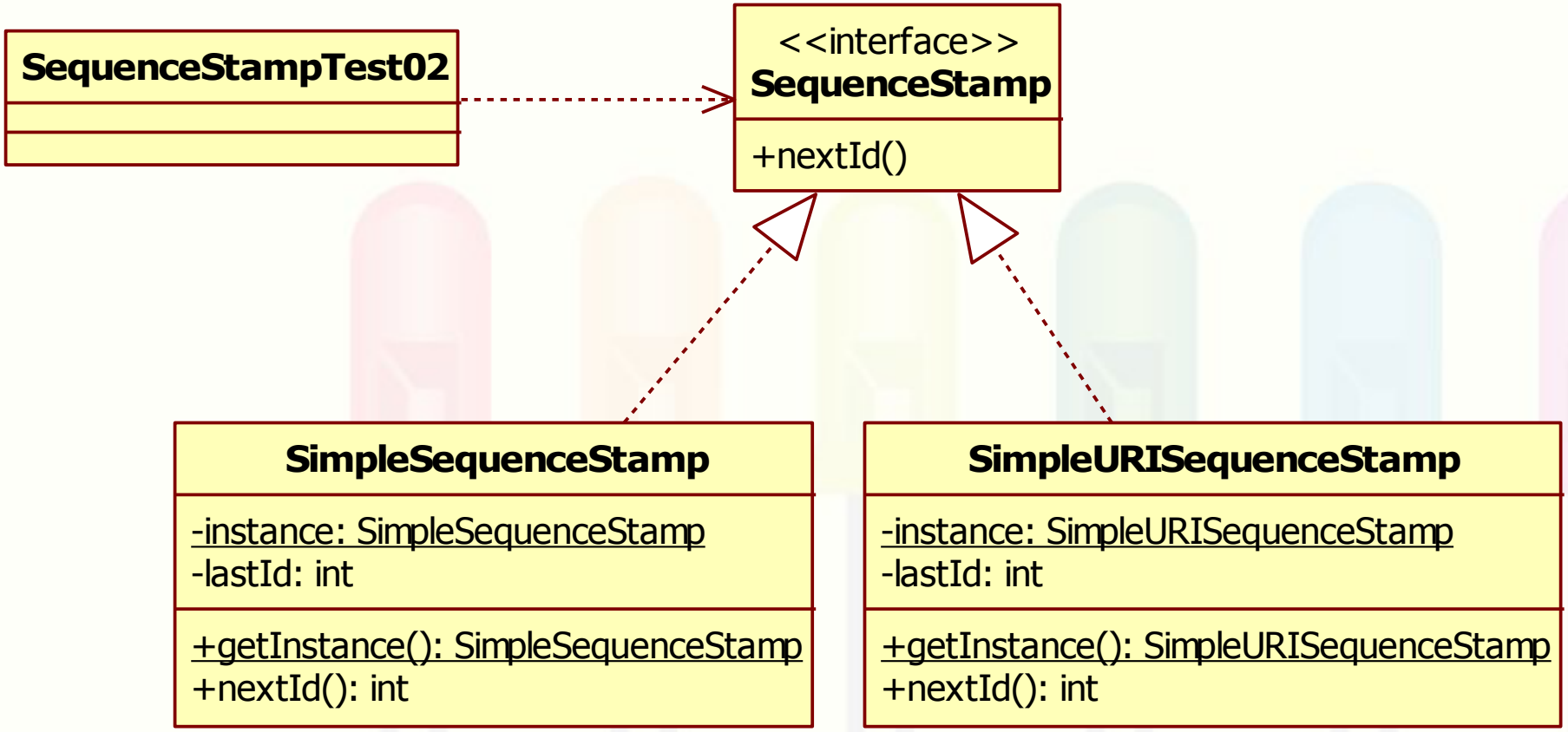
Aplicação do *Dependency Inversion Principle*

Dependency Inversion Principle (DIP)

- “Depender das Abstrações. Não depender das Concretizações.” (Martin, 2000)

Segunda Versão

Aplicação do *Dependency Inversion Principle*

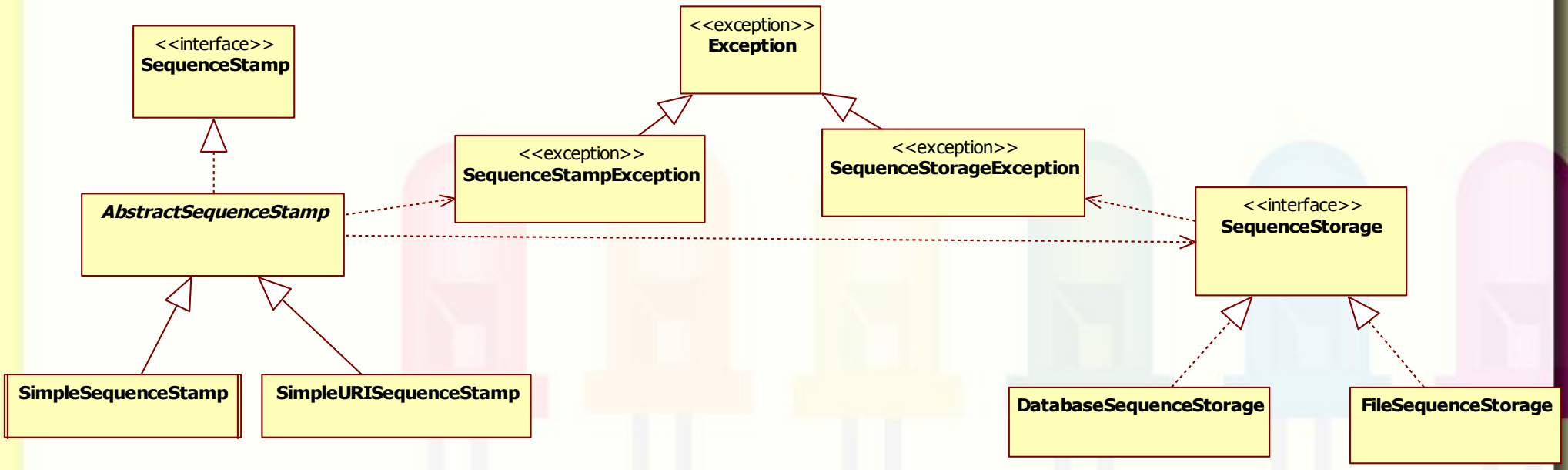


Componentização Sucessiva
Terceira Versão

Acrescentando capacidade de armazenamento

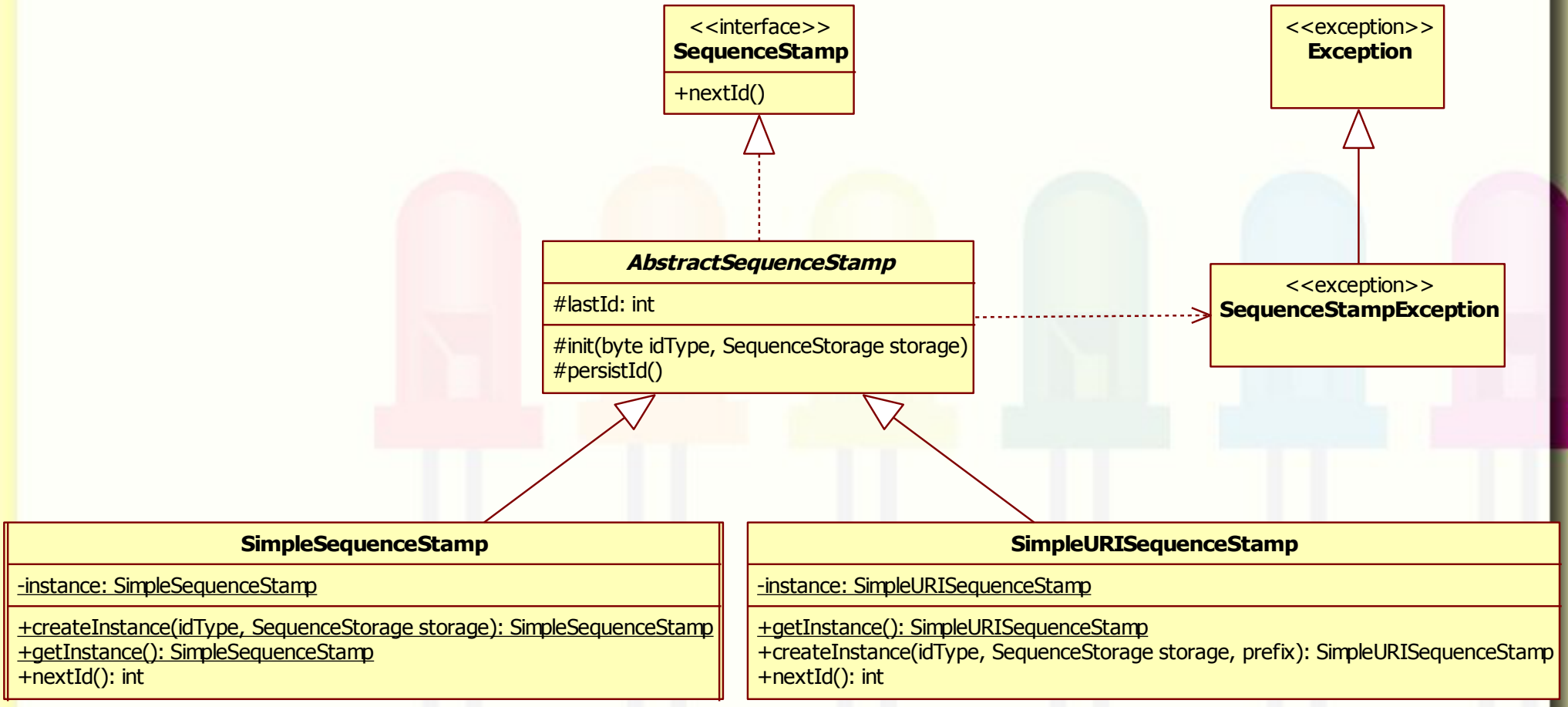
Mini *Framework*

Terceira Versão
Mini *Framework*



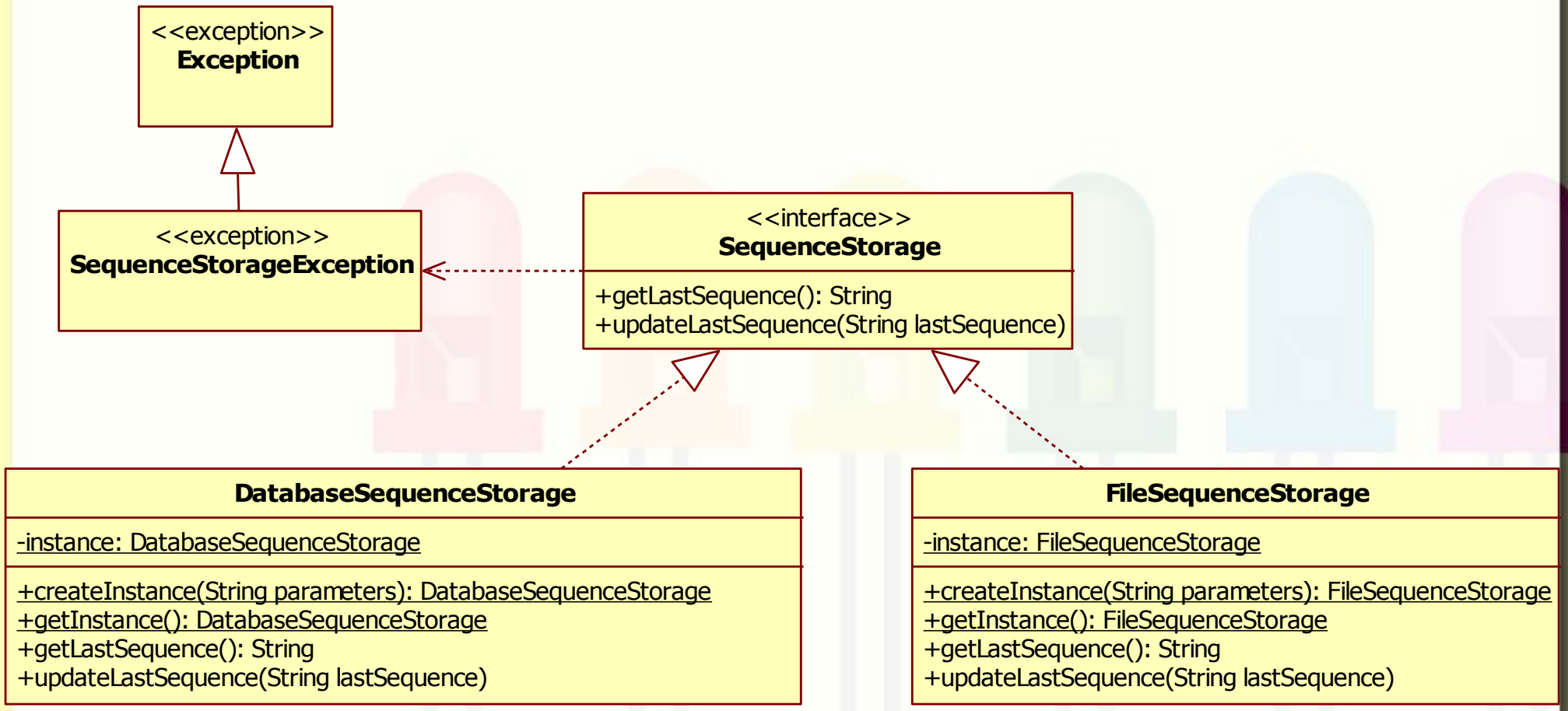
Terceira Versão

Detalhamento SequenceStamp



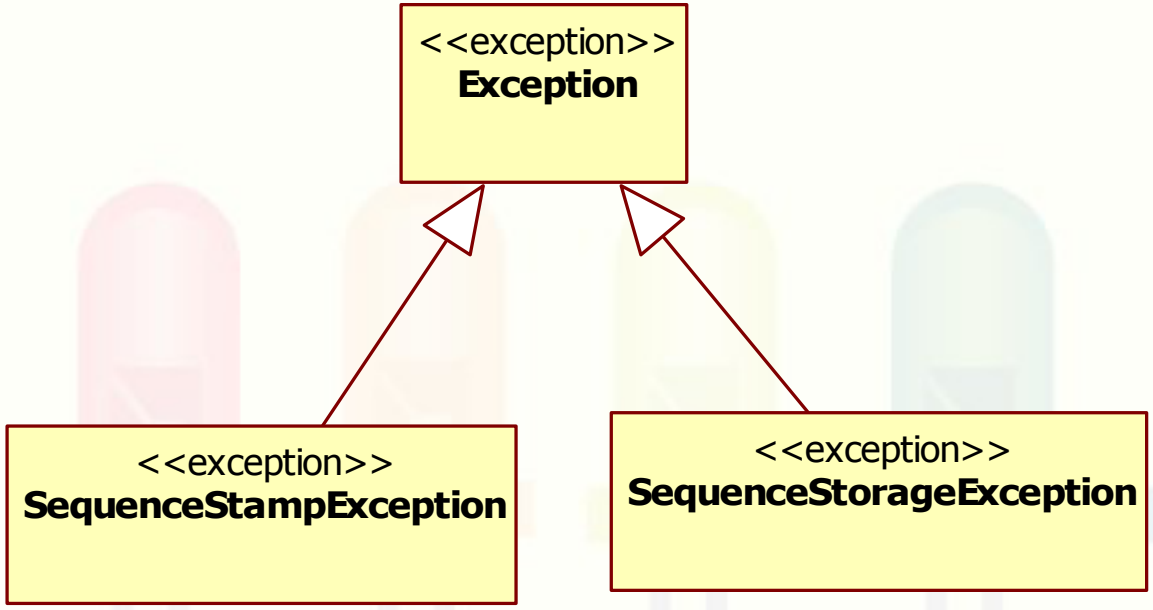
Terceira Versão

Detalhamento SequenceStorage



Terceira Versão

Detalhamento Exceptions



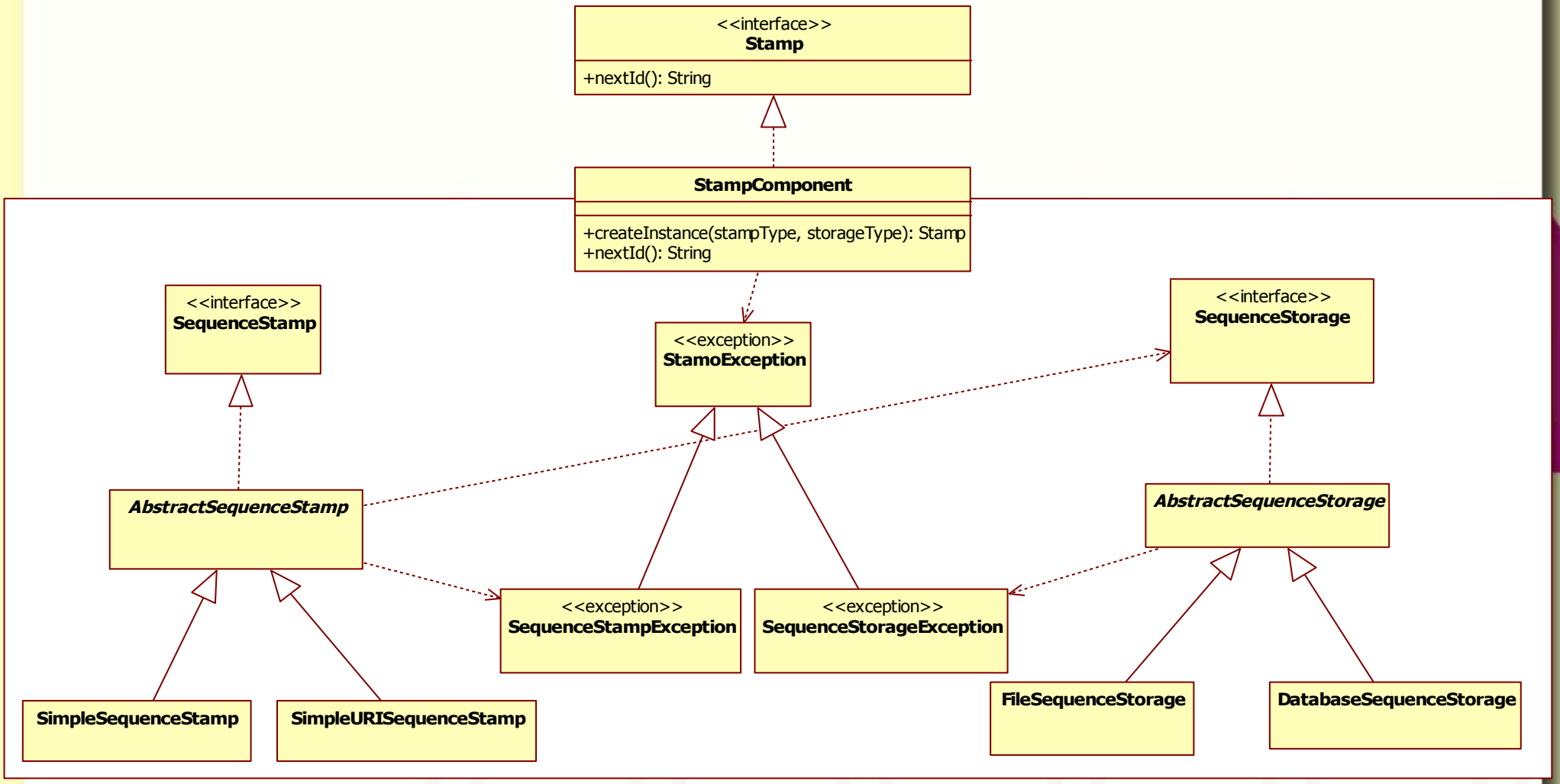
Componentização Sucessiva

Quarta Versão

Interface única para componente

Usando o padrão *Facade*

Classes e Componentes
Quarta Versão



Digital Content Component (DCC)

Fish DCC

- Goal
  - Draw a character-based Fish

```
 _
|\/o\
|/\_/
```

Step 1
Modeling

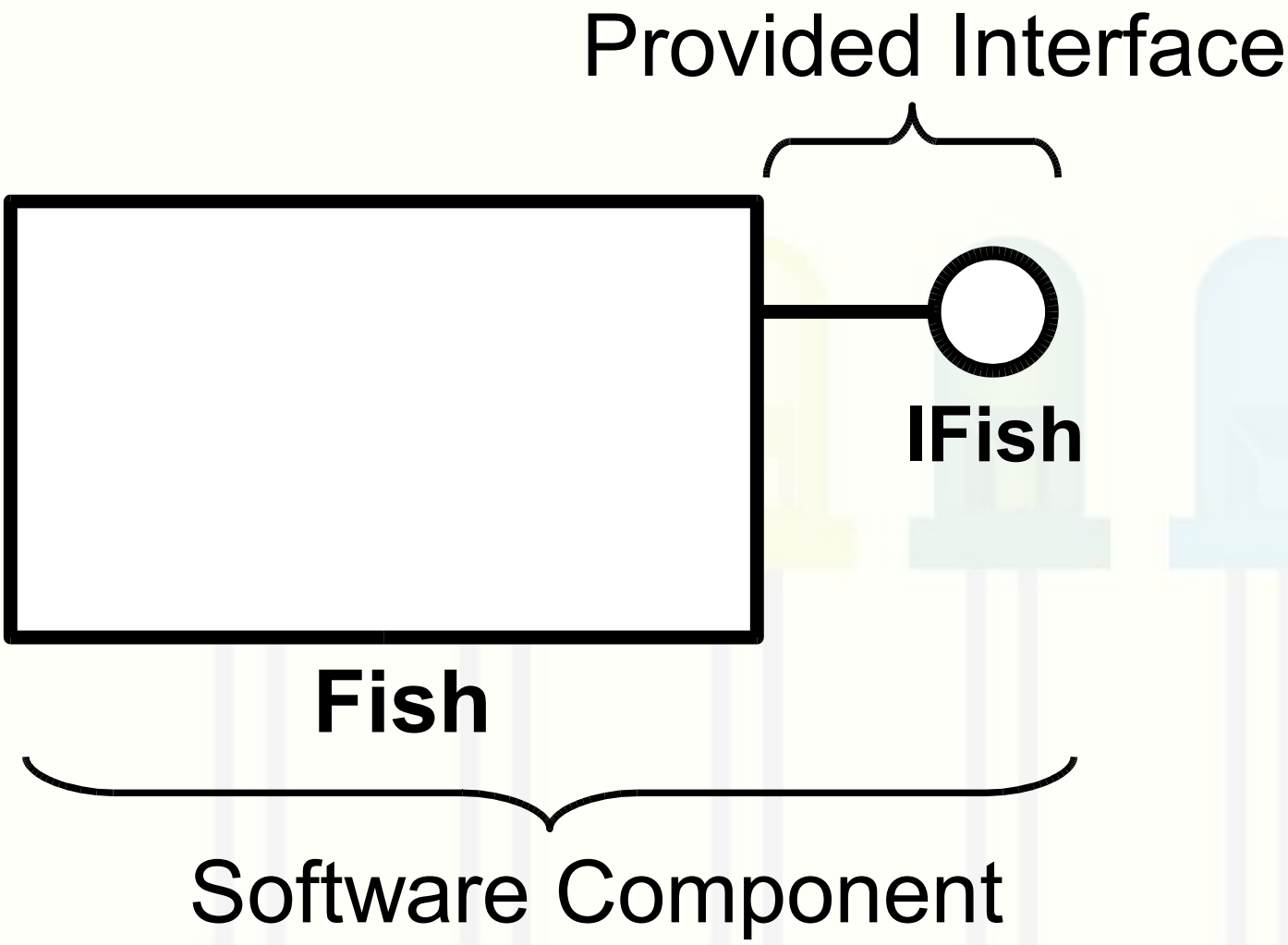
DCC Principle

- Publicly available DCC methods are exclusively accessed through DCC interfaces

Provided Interface

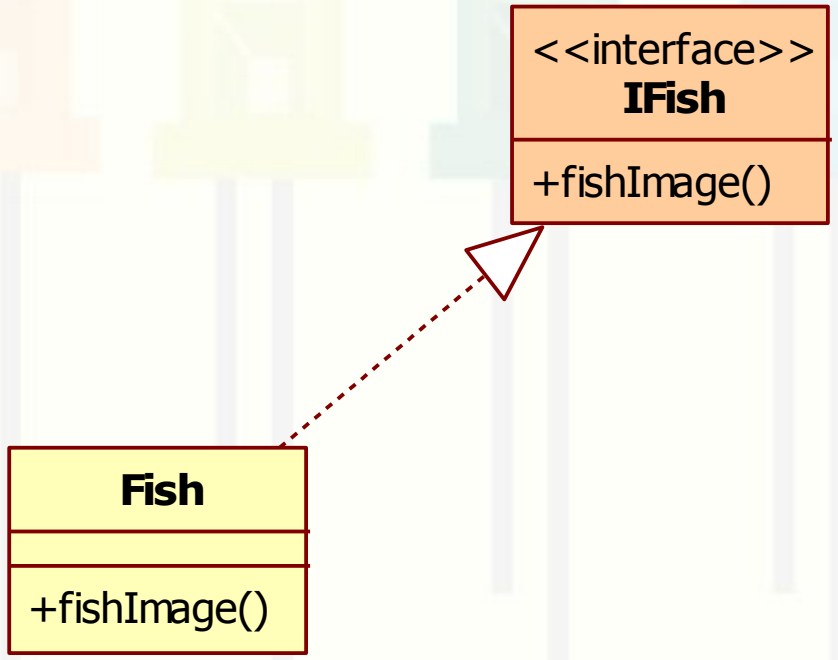
- Specifies services provided by a component

Component and Interface

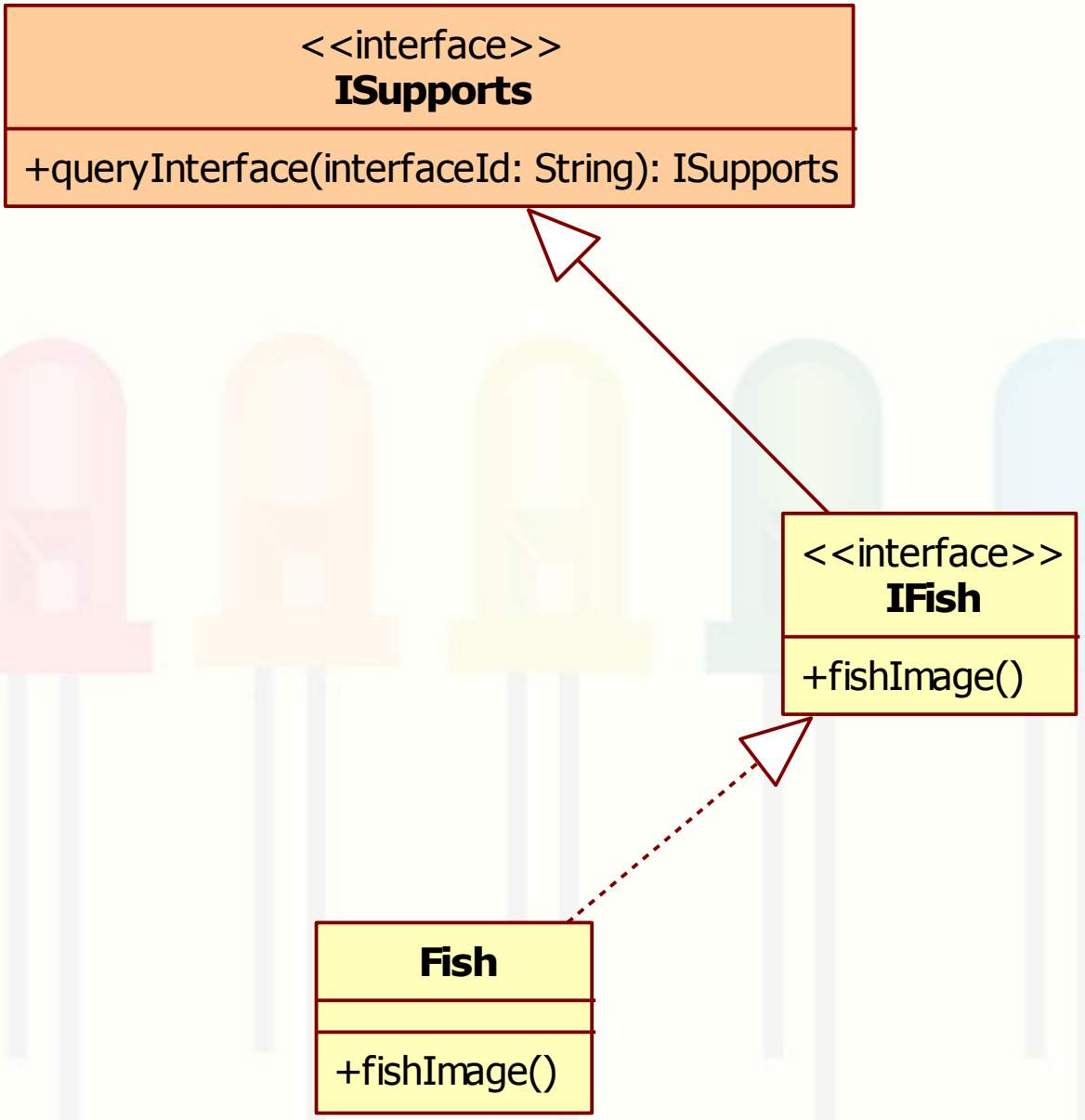


Fish Modeling

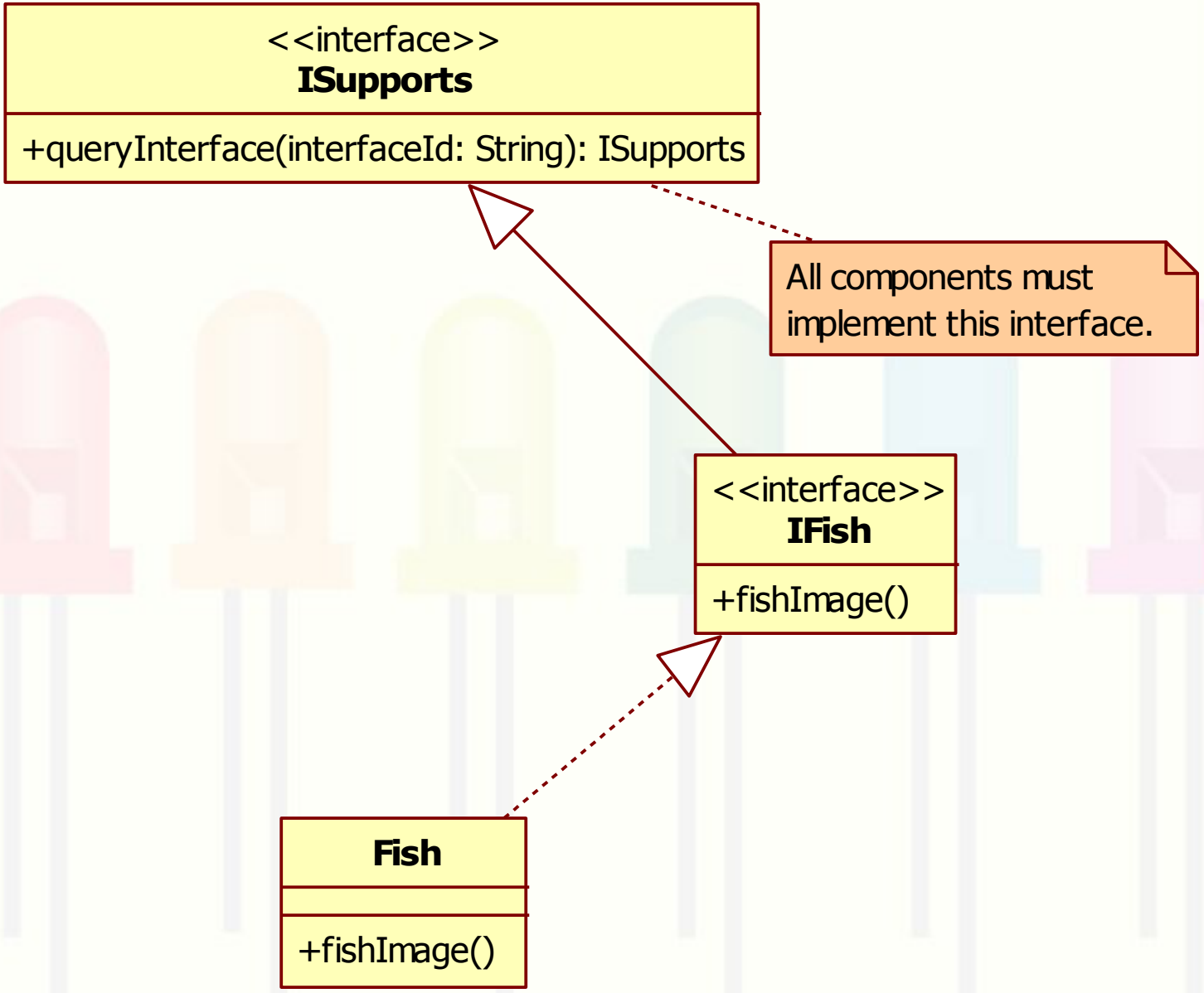
Fish Modeling



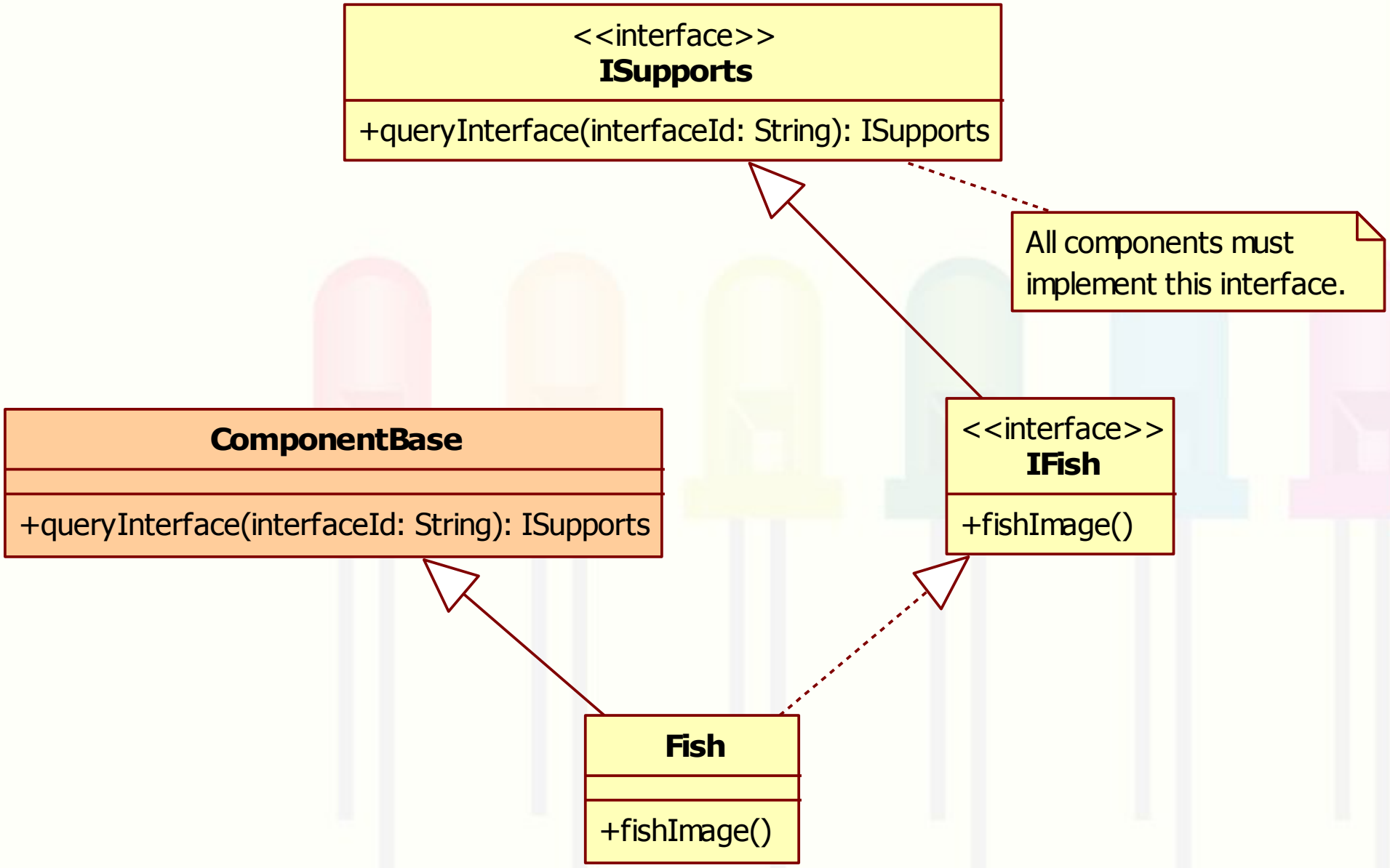
Fish Modeling



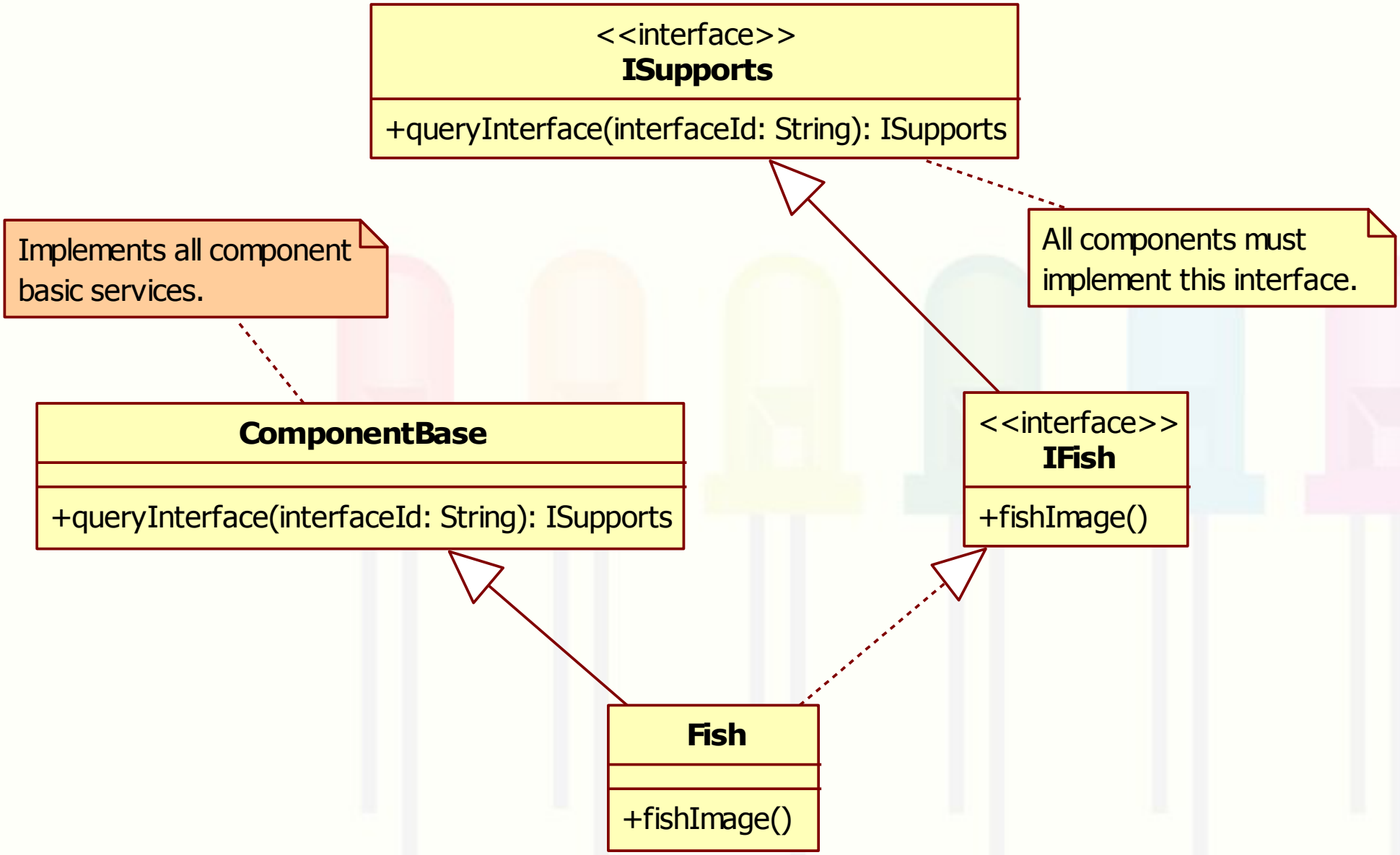
Fish Modeling



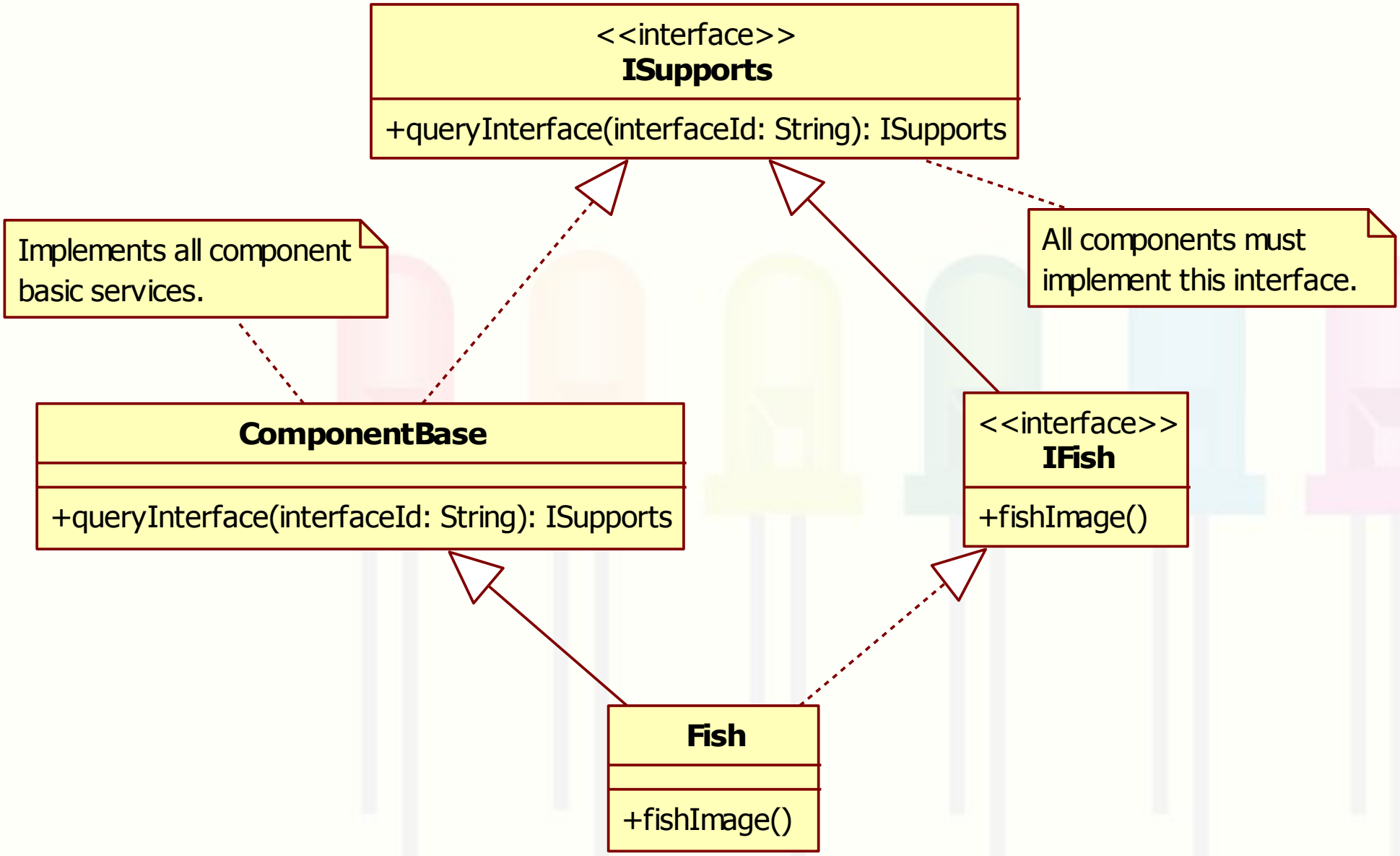
Fish Modeling



Fish Modeling



Fish Modeling



Step 2
Identifying

URI-based Identification

- DCC identification approach is based on URIs
- See details in

http://apps.sourceforge.net/mediawiki/infrabig/index.php?title=DCC_Identification

Creating an Identification

- URI prefix + Class path
- Ex.:

1) infraBig/DCC URI prefix:

`http://purl.org/NET/dcc/`

2) Component class path:

`examples.fish.s01.Fish`

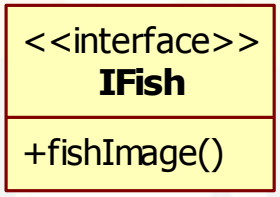
3) Result:

`http://purl.org/NET/dcc/examples.fish.s01.Fish`

Step 3
Documenting

Interface Card

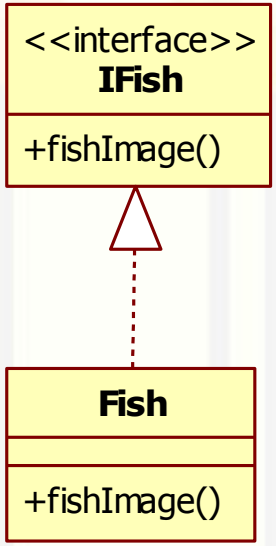
Title	Fish Interface
Id	`http://purl.org/NET/dcc/examples.fish.s01.IFish`
Author	André Santanchè
Goal	Interface for the Fish component that enables to trigger the fish drawing.
Methods	
fishImage	Draw the fish.
UML Diagram	



Component Card

Title	Fish Component
Id	`http://purl.org/NET/dcc/examples.fish.s01.Fish`
Author	André Santanchè
Goal	Draw a character-based Fish.
Provided Interfaces	Fish Interface `http://purl.org/NET/dcc/examples.fish.s01.IFish`

UML Diagram



Step 4
Implementing

Cards to Components

IFish

Author	André Santanchè
Goal	Interface for the Fish component that enables to trigger the fish drawing.

```
/**
 * Interface for the Fish component that
 * enables to trigger the fish drawing.
 *
 * @author Andre Santanche
 *
 */

public interface IFish extends ISupports
```

Cards to Components

IFish

Id	http://purl.org/NET/dcc/examples.fish.s01.IFish

```
@ComponentInterface(
    "http://purl.org/NET/dcc/examples.fish.s01.IFish")

public interface IFish extends ISupports
```

Cards to Components

IFish

fishImage	Draw the fish.

```
/**
 * Draw the fish.
 */

public String fishImage();
```

Cards to Components

Fish

Id	http://purl.org/NET/dcc/examples.fish.s01.Fish

Provided Interfaces	Fish Interface http://purl.org/NET/dcc/examples.fish.s01.IFish

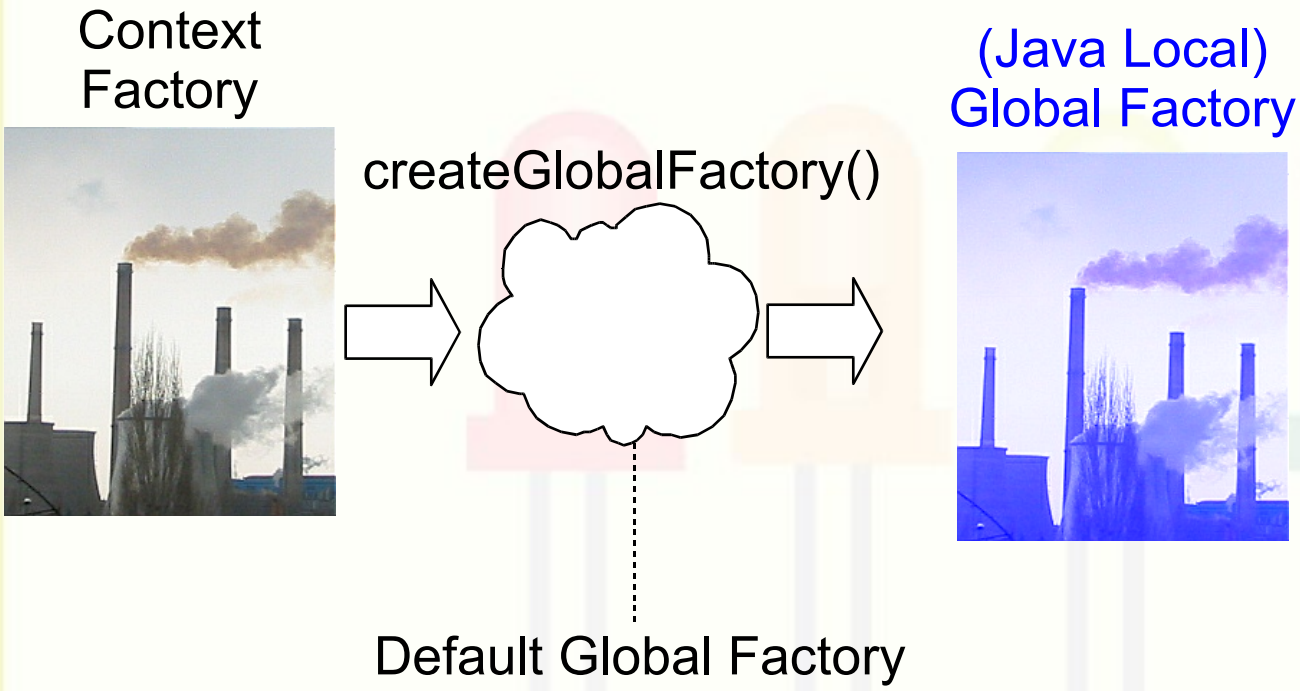
```
@Component(
   id="http://purl.org/NET/dcc/examples.fish.s01.Fish",
   provides={"http://purl.org/NET/dcc/examples.fish.s01.IFish"}
)
```

Step 5
Using

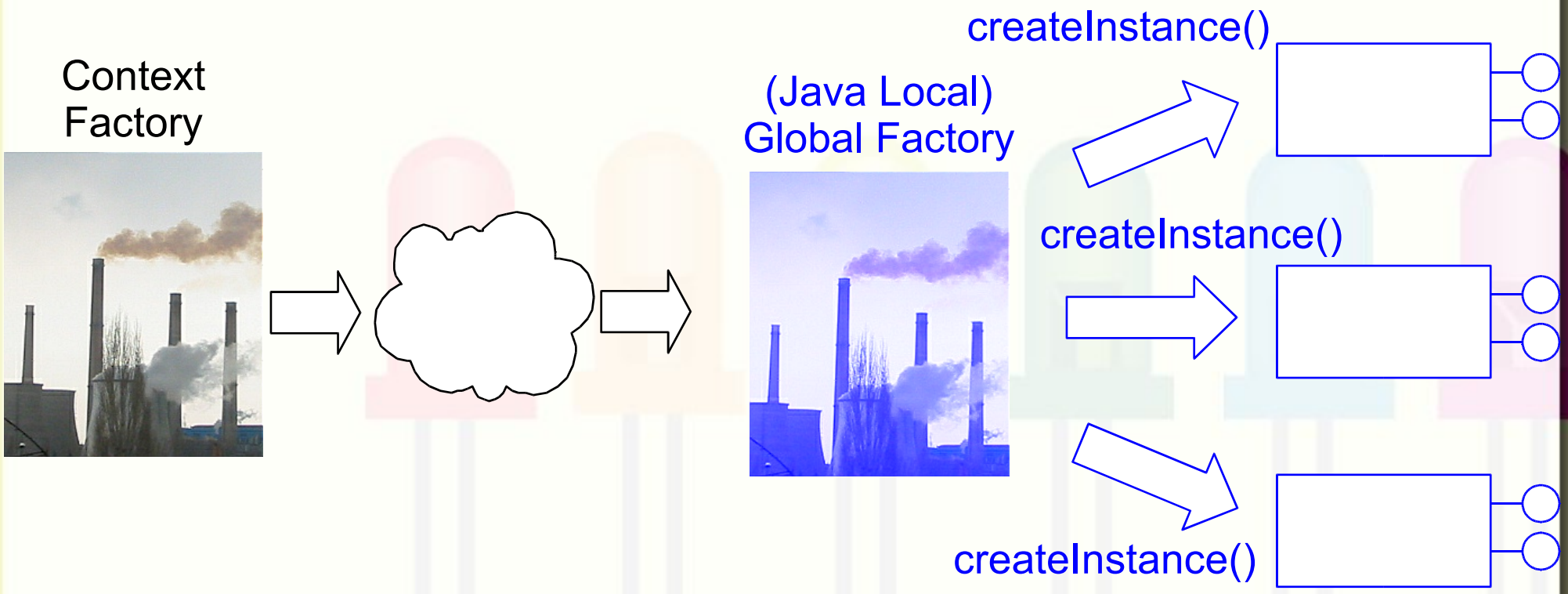
Instantiation and Abstract Factory

- DCCs are instantiated using the Abstract Factory Design Pattern
- See detailed description in:
  - Gamma, E. Helm, R. Johnson, R. Vlissides, J. **Design Patterns: Elements of Reusable Object-Oriented Software**. Addison-Wesley, 1995.

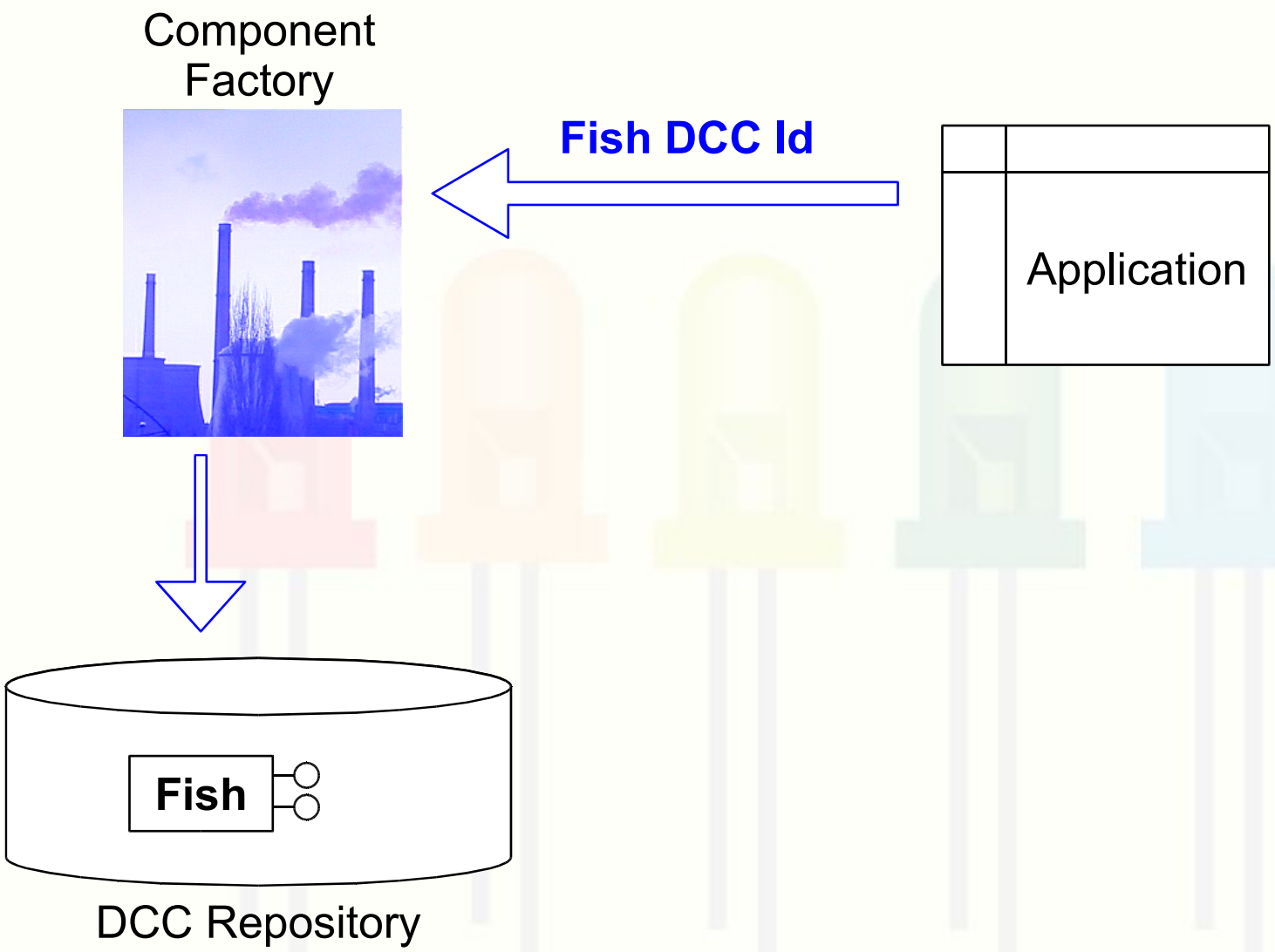
Creating a Global Factory



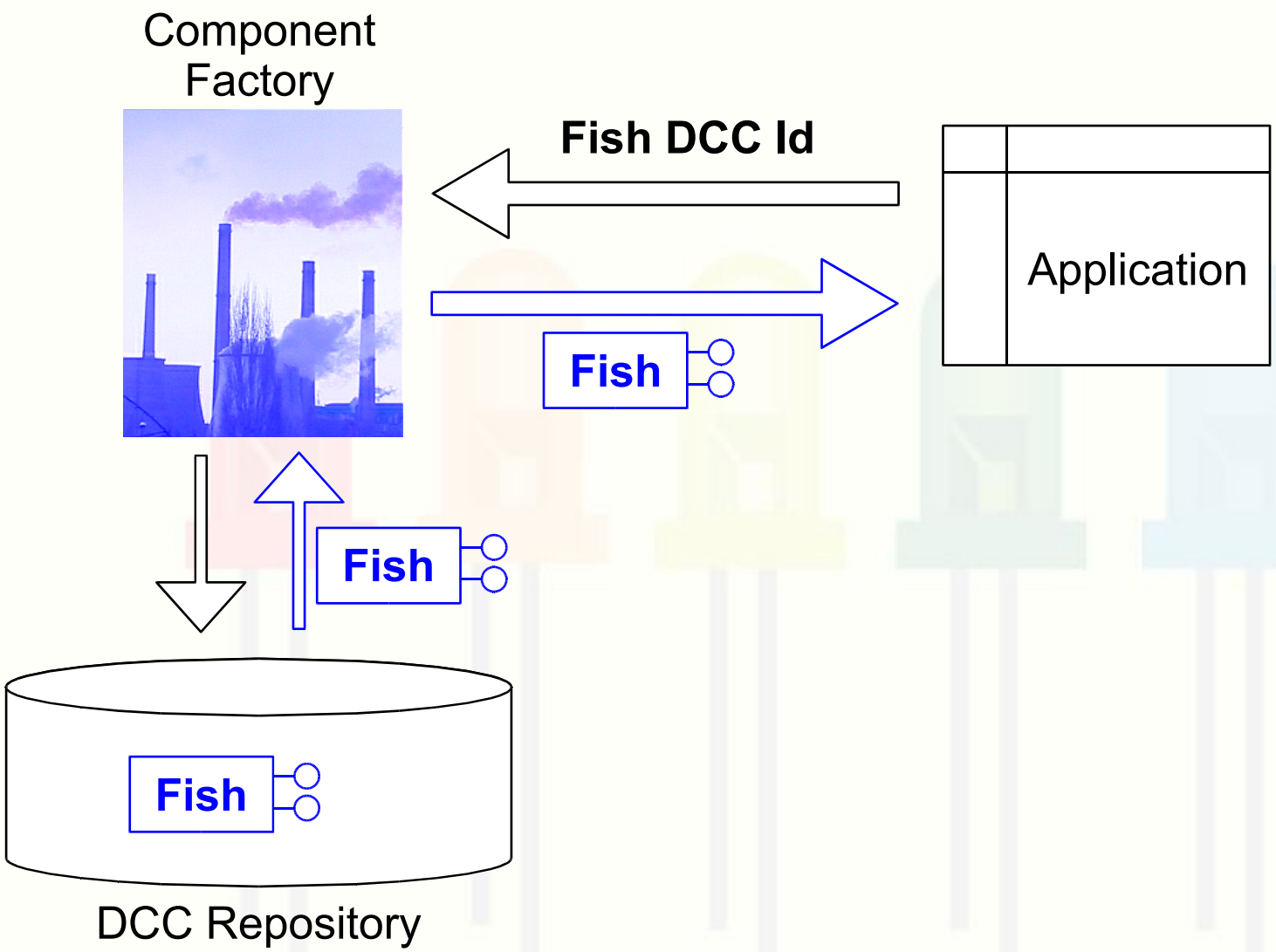
Creating DCCs using the Factory



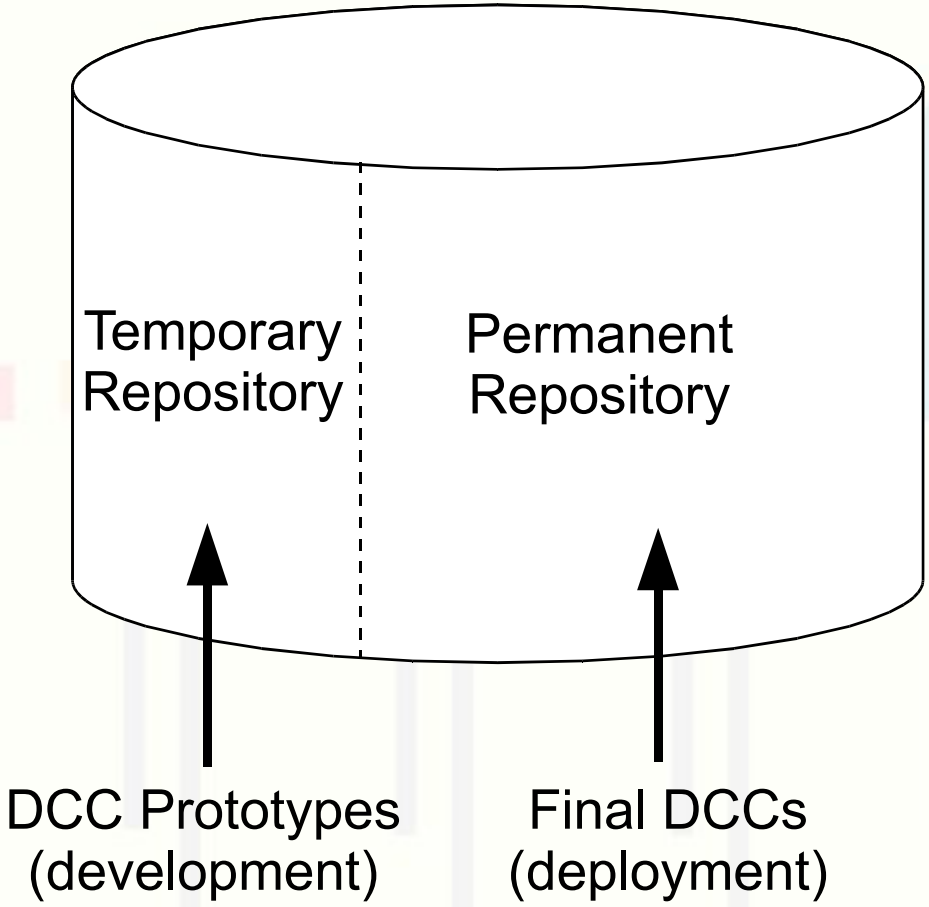
DCC Repository



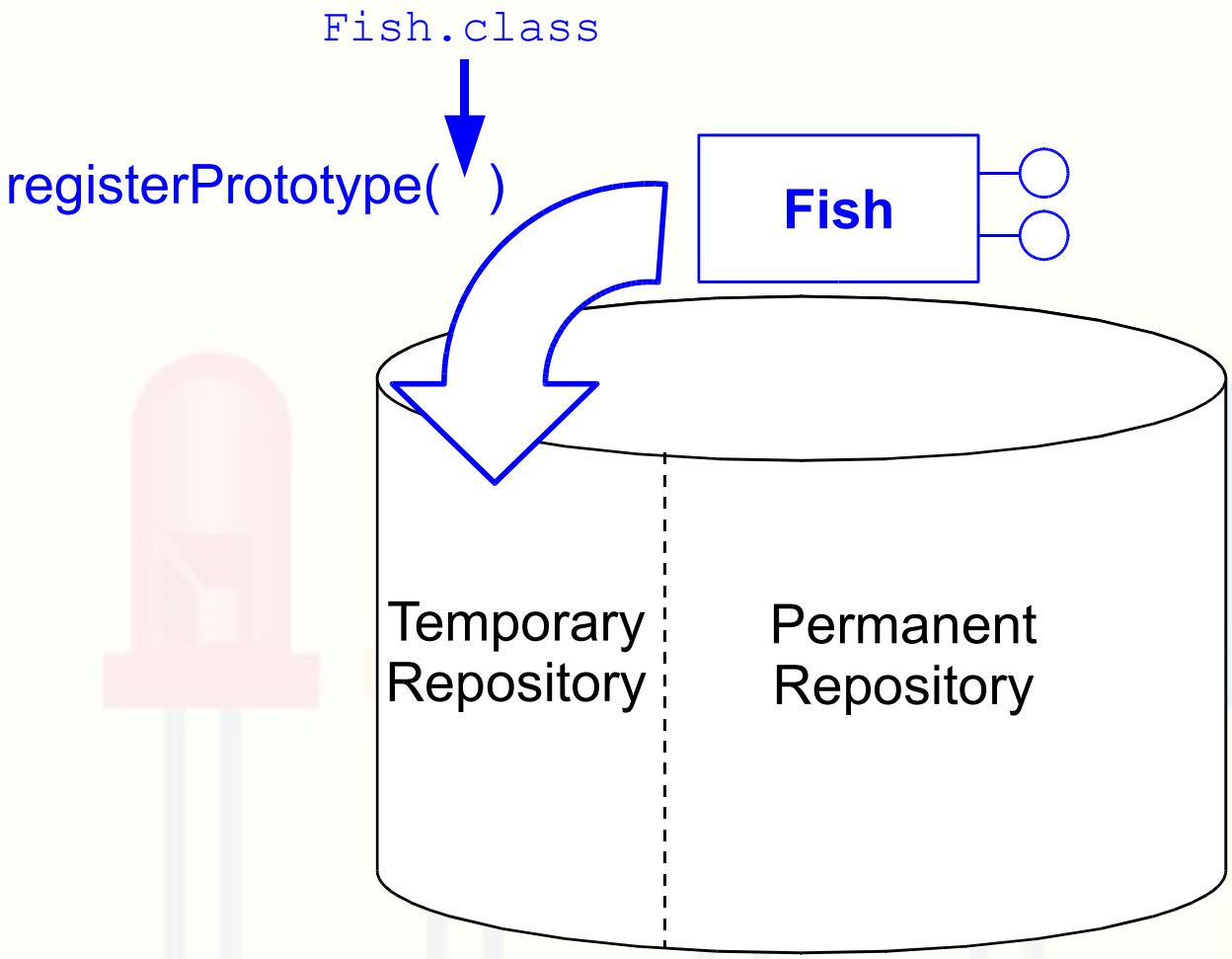
DCC Repository



Temporary Repository

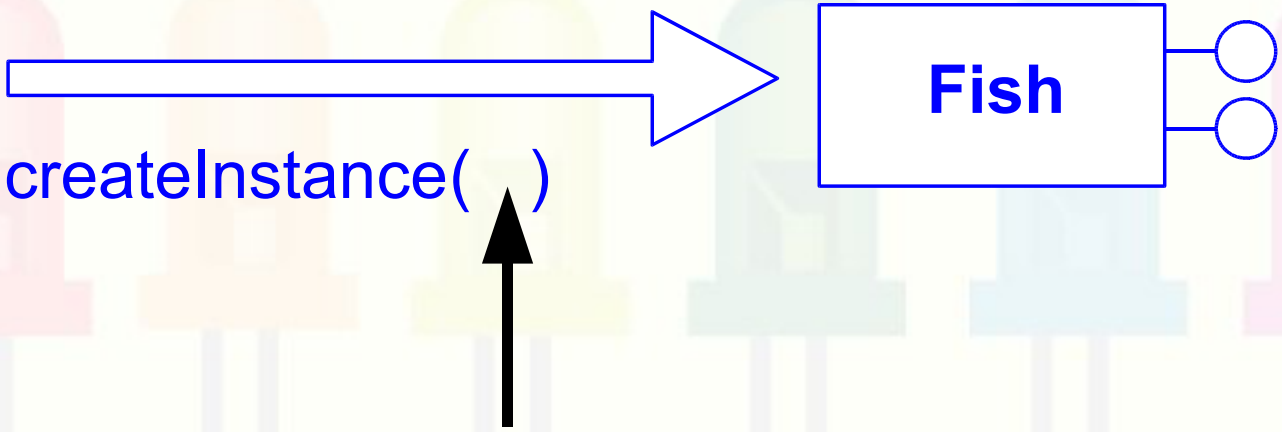


Temporary Repository



Creating a DCC using the Factory

Component Factory



`http://purl.org/NET/dcc/examples.fish.s01.Fish`

Fish DCC Id

Objetivo do DCC

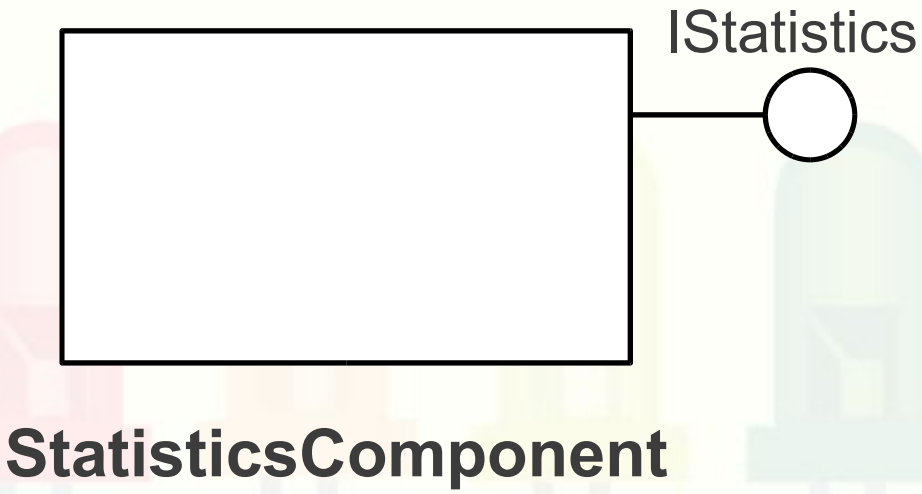
- Registrar um conjunto de números e calcular a soma e média destes números.

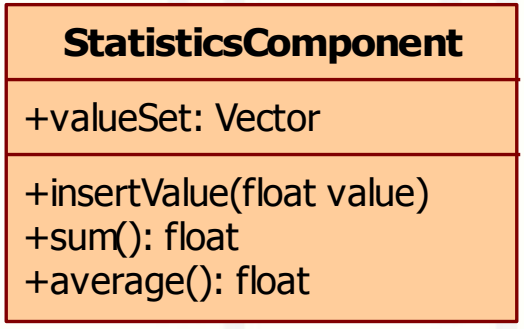
Delimitação

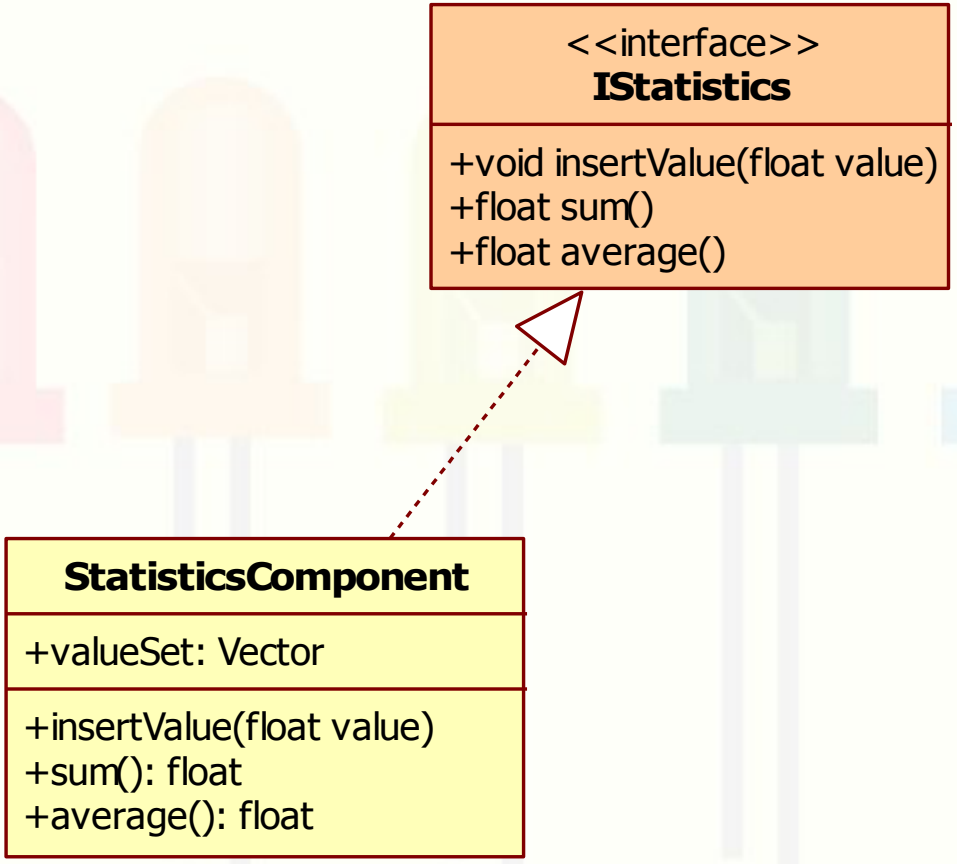
- DCC deve ter delimitações explícitas
  - Essencial para distribuição e reuso
  - Estratégia básica: único pacote
- Pacote do componente de estatísticas:
  - `pt.c02foundations.statistics.s01`

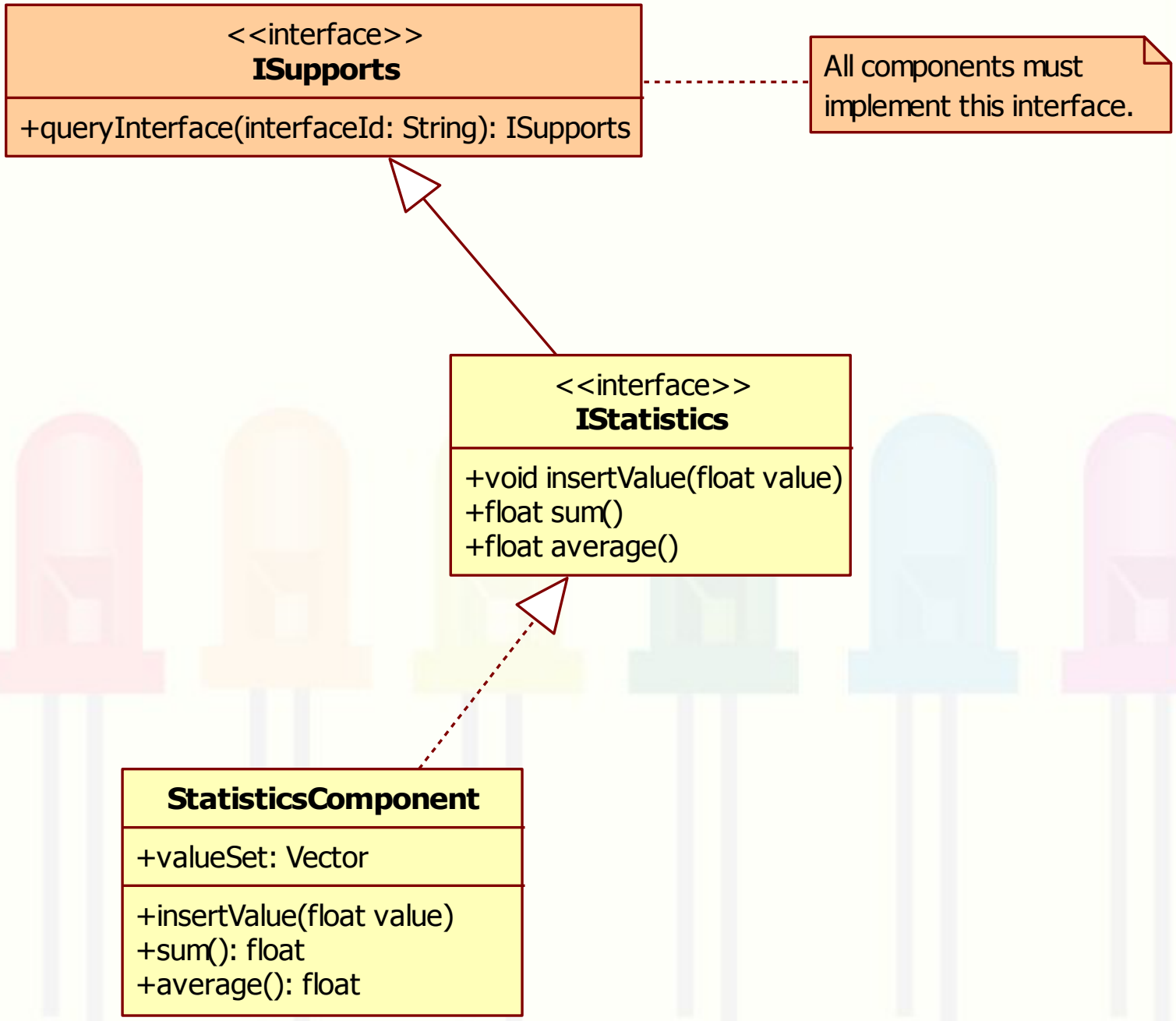
Projetando o DCC

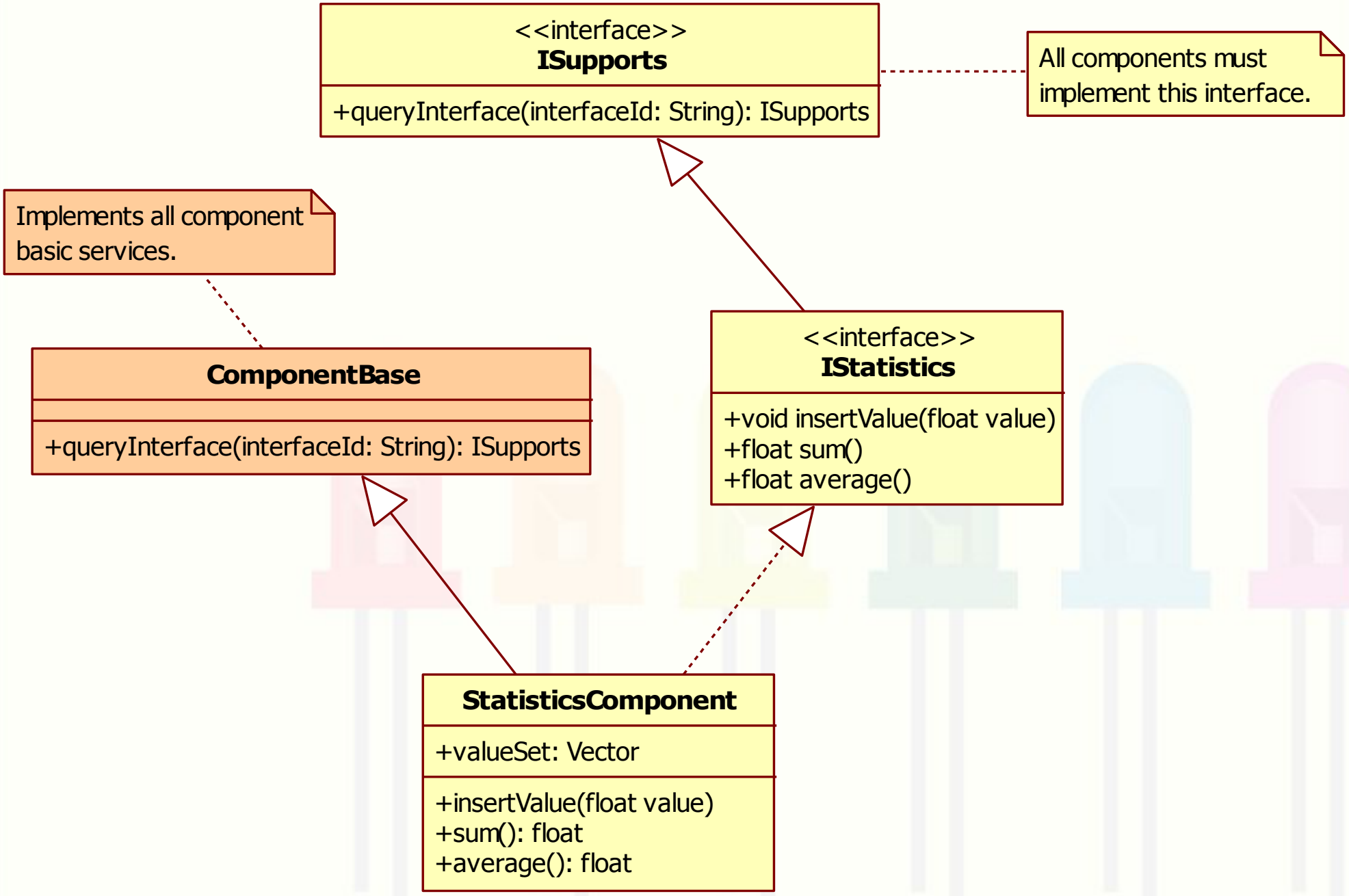
Componente e Interface Provida

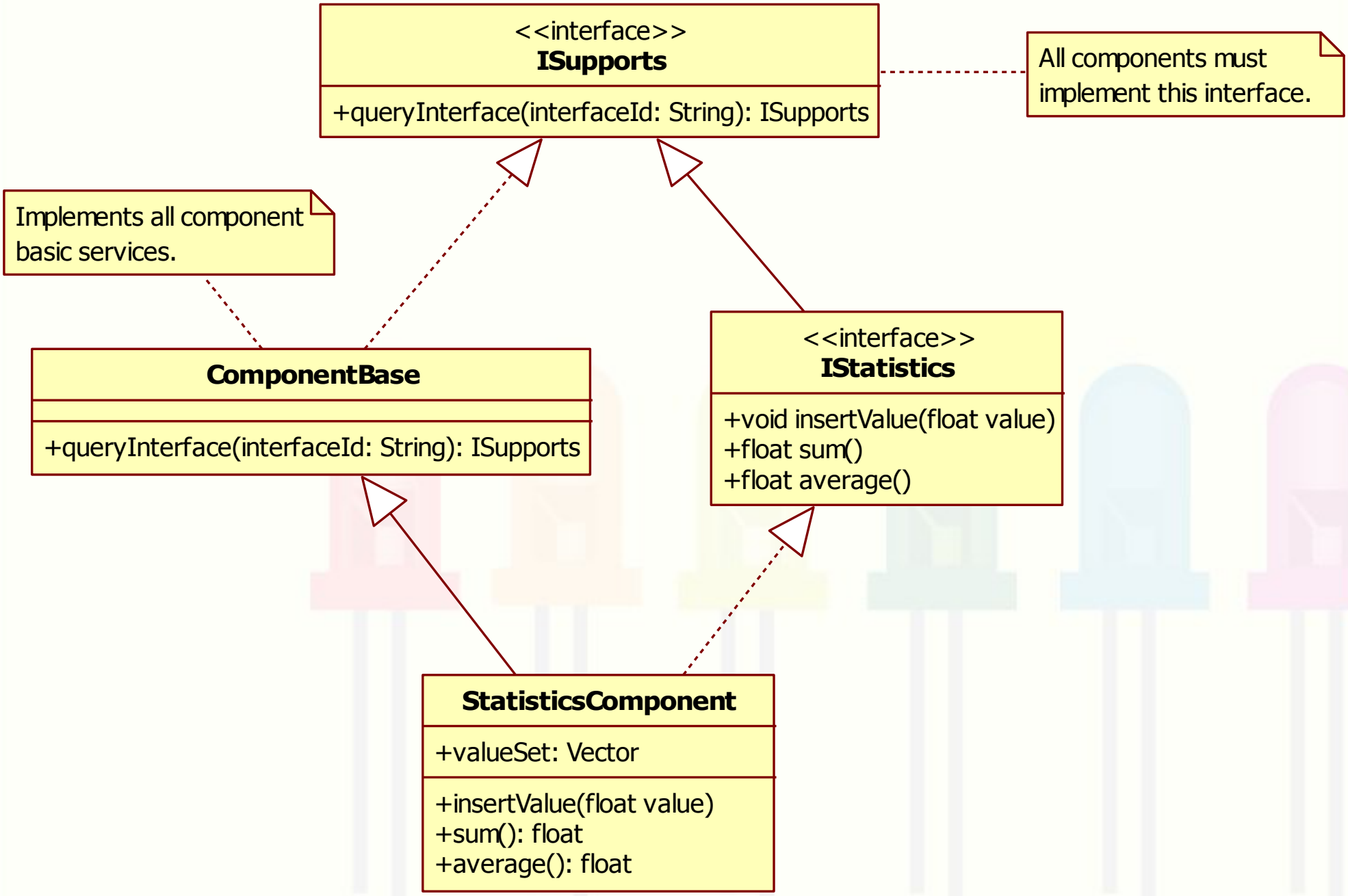



StatisticsComponent
+valueSet: Vector
+insertValue(float value)
+sum(): float
+average(): float


<<interface>>
IStatistics
+void insertValue(float value)
+float sum()
+float average()
StatisticsComponent
+valueSet: Vector
+insertValue(float value)
+sum(): float
+average(): float


<<interface>>
ISupports
+queryInterface(interfaceId: String): ISupports
All components must implement this interface.
<<interface>>
IStatistics
+void insertValue(float value)
+float sum()
+float average()
StatisticsComponent
+valueSet: Vector
+insertValue(float value)
+sum(): float
+average(): float


<<interface>>
ISupports
+queryInterface(interfaceId: String): ISupports
All components must implement this interface.
Implements all component basic services.
ComponentBase
+queryInterface(interfaceId: String): ISupports
<<interface>>
IStatistics
+void insertValue(float value)
+float sum()
+float average()
StatisticsComponent
+valueSet: Vector
+insertValue(float value)
+sum(): float
+average(): float


<<interface>>
ISupports
+queryInterface(interfaceId: String): ISupports
All components must implement this interface.
Implements all component basic services.
ComponentBase
+queryInterface(interfaceId: String): ISupports
<<interface>>
IStatistics
+void insertValue(float value)
+float sum()
+float average()
StatisticsComponent
+valueSet: Vector
+insertValue(float value)
+sum(): float
+average(): float

Criando uma Identificação

- URI prefixo + caminho da Classe
- Ex.:

1) DCC URI Namespace:

`http://purl.org/dcc/`

2) Caminho da classe do componente:

`pt.c02foundations.statistics.s01.StatisticsComponent`

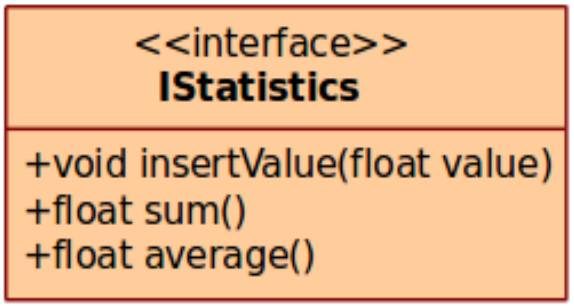
3) Resultado:

`http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics`

Documentação

Interface Specification

Title	Statistics Interface	
Id	<http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics>	
Author	André Santanchè	
Description	Interface for a Statistics Component that registers a set of numbers and calculates the sum and average of these numbers.	
Methods		
insertValue	Insert a value into the set.	
	value	the value to be inserted into the set
sum	Return the sum of the values in the set. Return zero if the set is empty.	
	return	sum of the values in the set
average	Return the average of the values in the set. Return zero if the set is empty.	
	return	average of the values in the set
UML Diagram		

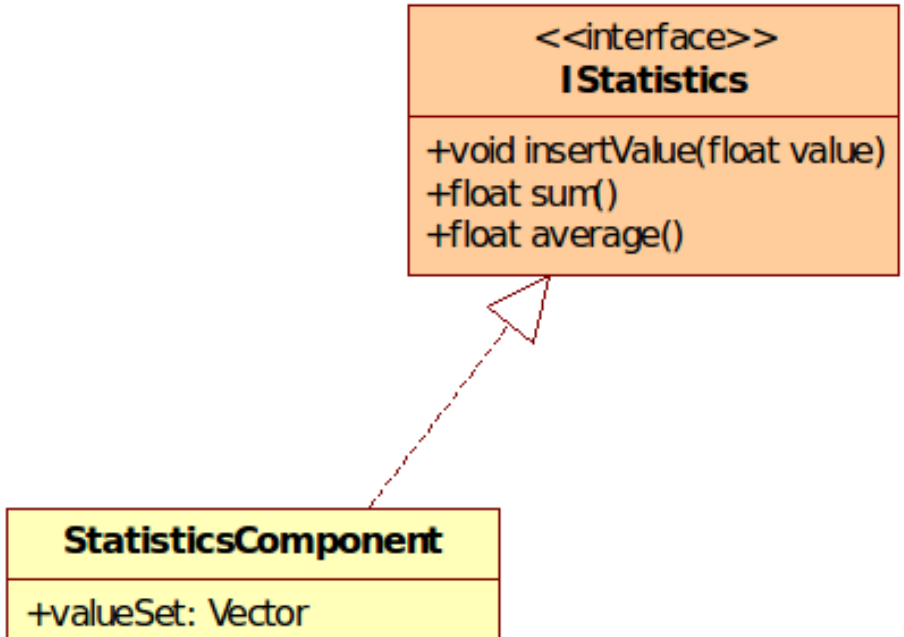


Component Specification

Title	Statistics Component
Id	<http://purl.org/dcc/pt.c02foundations.statistics.s01.StatisticsComponent>
Author	André Santanchè
Description	Registers a set of numbers and calculates the sum and average of these numbers.
Provided Interfaces	

Title	**Statistics Interface**
Id	<http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics>

UML Diagram



Codificação em Java

Da Ficha ao Componente
IStatistics

Author	André Santanchè
Description	Interface for a Statistics Component that registers a set of numbers and calculates the sum and average of these numbers.

```
/**
 * Interface for a Statistics Component that
   registers a set of numbers
 * and calculates the sum and average of these
   numbers.
 *
 * @author Andre Santanche
 */
public interface IStatistics extends ISupports
```

Da Ficha ao Componente
IStatistics

Id	<http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics>

```
@ComponentInterface(
  "<http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics>"
)

public interface IStatistics extends ISupports
```

Da Ficha ao Componente
IStatistics

insertValue	Insert a value into the set.	
	value	the value to be inserted into the set

```
/**
 * Insert a value into the set.
 * @param value the value to be inserted into
   the set
 */
public void insertValue(float value);
```

Da Ficha ao Componente
StatisticsComponent

Id	<http://purl.org/dcc/pt.c02foundations.statistics.s01.StatisticsComponent>

Provided Interfaces	

Title	Statistics Interface
Id	<http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics>

```
@Component(
 id =
  "<http://purl.org/dcc/pt.c02foundations.statistics.s01.StatisticsComponent>",
 provides =
  {"<http://purl.org/dcc/pt.c02foundations.statistics.s01.IStatistics>"}
)
```

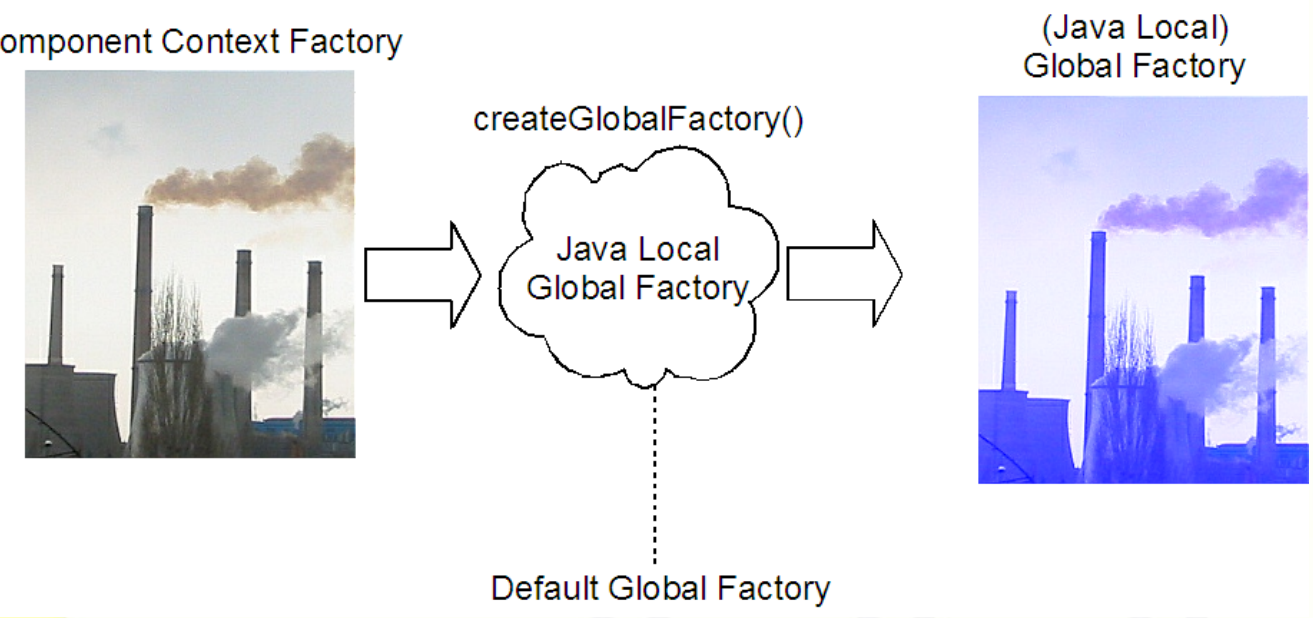
Caso 1

Primeira Versão

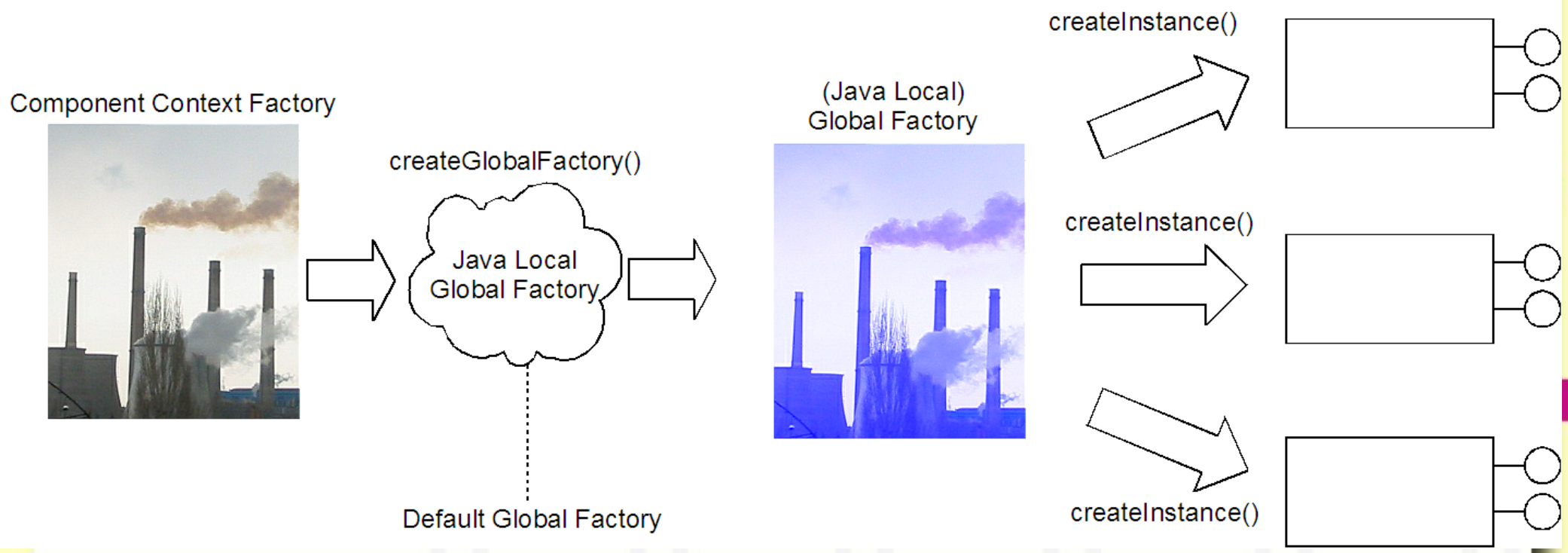
Statistics

Usando um Componente

Criação da Fábrica Global

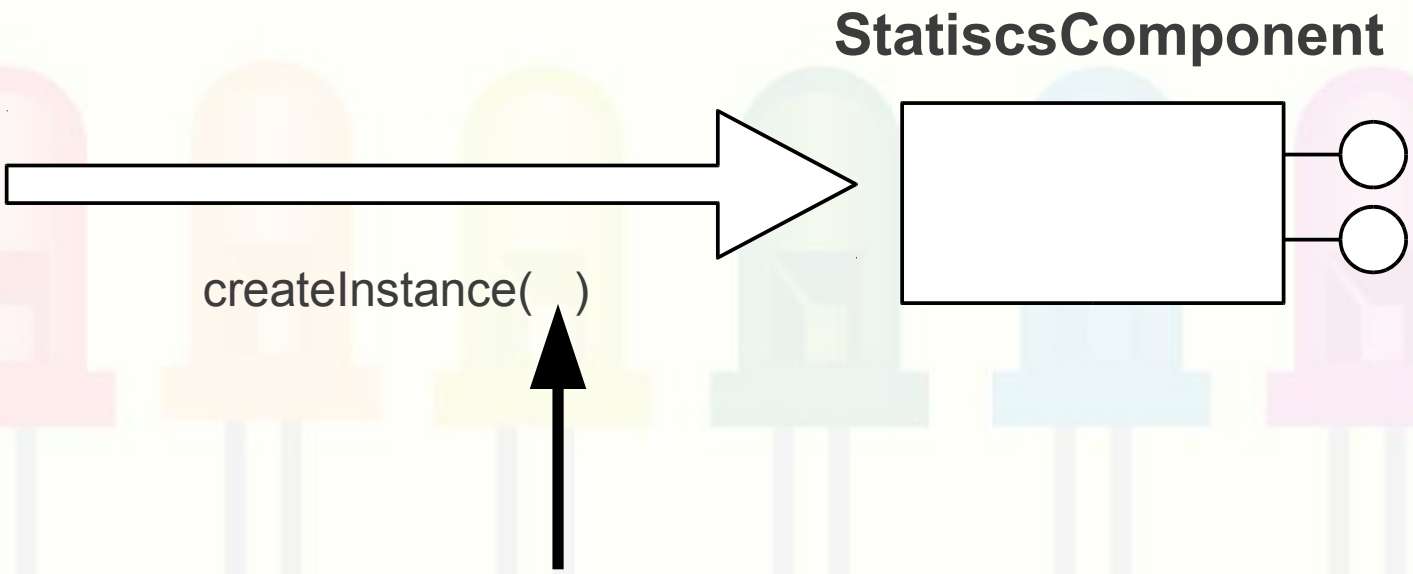


Criando componentes usando a fábrica



Criando componentes usando a fábrica

Component Factory



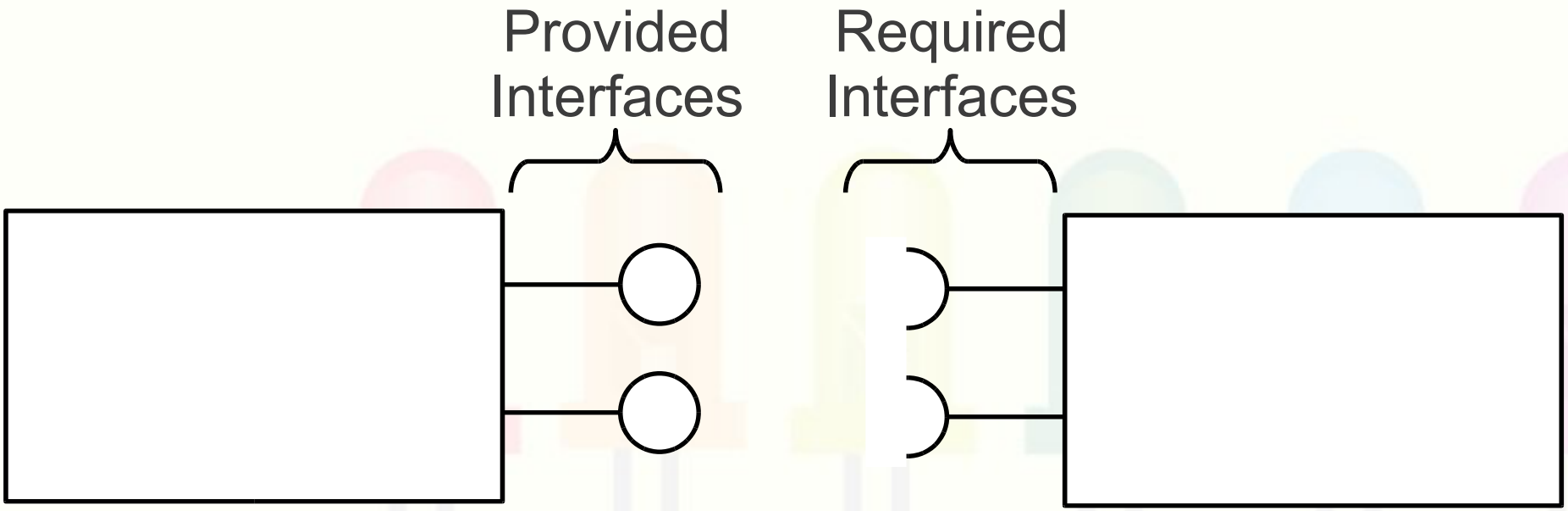
`<http://purl.org/dcc/pt.c02foundations.statistics.s01.StatisticsComponent>`

StatiscsComponent Id

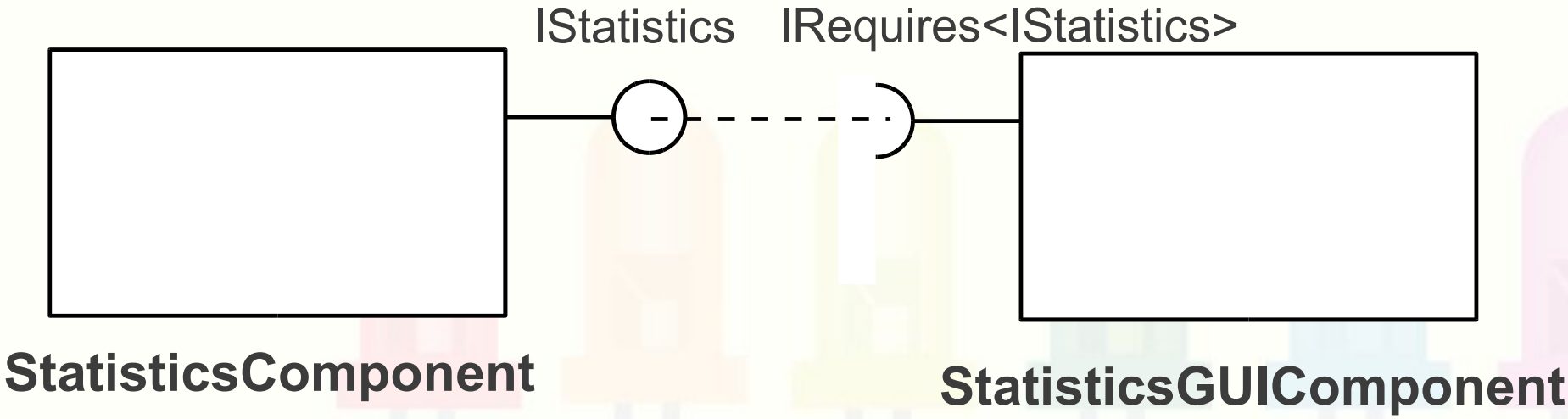
Case Study 2

Displaying Statistics

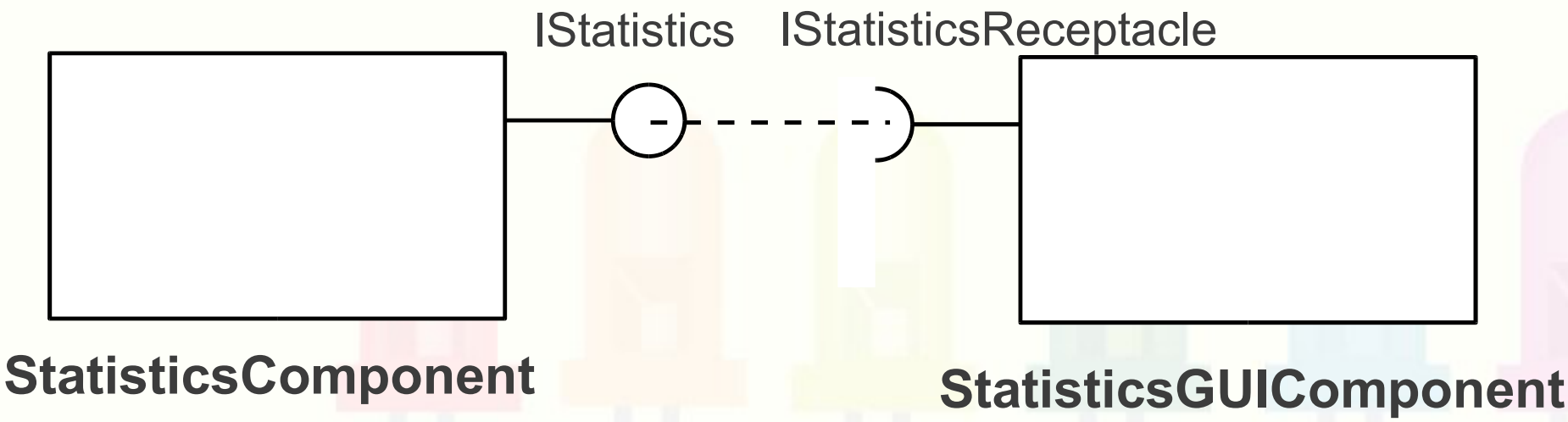
Interfaces Providas e Requeridas



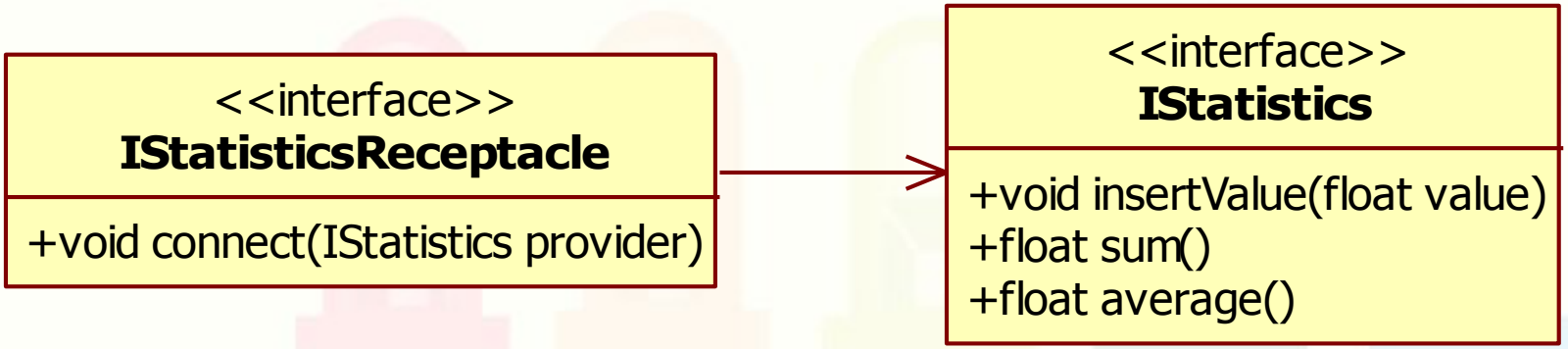
Interface Requerida
IStatisticsReceptacle



Interface Requerida
IStatisticsReceptacle



IStatisticsReceptacle



Bibliografia

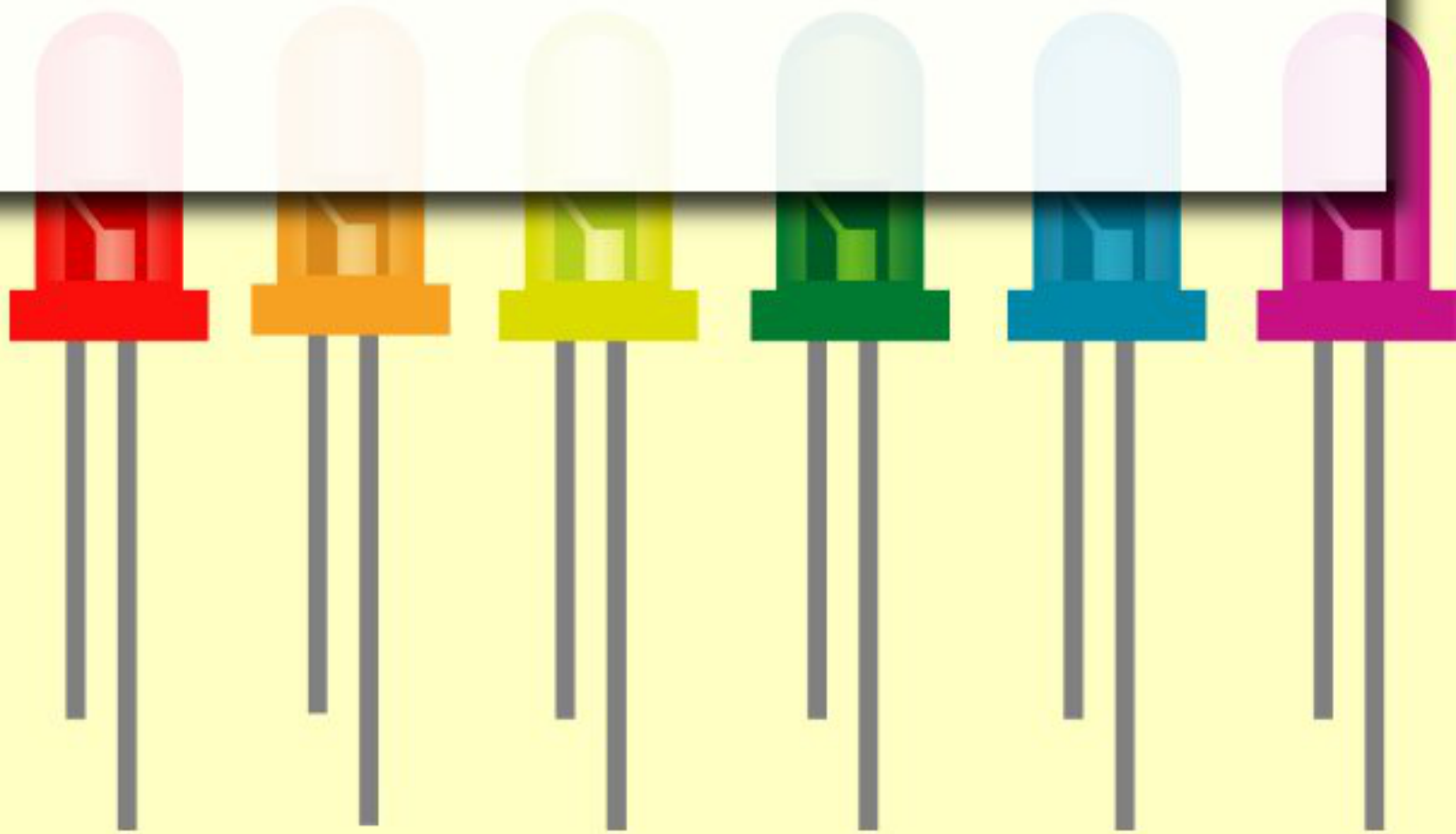
- Bachmann, F.; Bass, L.; Buhman, C.; Dorda, S.C.; Long, F.; Robert, J. & Wallnau, R.S.K. **Volume II: Technical Concepts of Component-Based Software Engineering**, 2nd Edition. Carnegie Mellon University, 2000.
- Broy, M.; Deimel, A.; Henn, J.; Koskimies, K.; Plásil, F.; Pomberger, G.; Pree, W.; Stal, M. & Szyperski, C. **What characterizes a (software) component?** Software -- Concepts & Tools, Springer-Verlag Heidelberg, 1998, 19, 49-56.
- Gamma, E. Helm, R. Johnson, R. Vlissides, J. **Design Patterns: Elements of Reusable Object-Oriented Software**. Addison-Wesley, 1995.

Bibliografia

- Hopkins, J. **Component primer**. Communications ACM, ACM Press, 2000, 43, 27-30.
- Martin, R. C. **Design Principles and Design Patterns**. Object Mentor, 2000.
- Mcilroy, M. D. Naur, P. & Randell, B. (ed.) **Mass Produced Software Components**. Software Engineering: Report of a conference sponsored by the NATO Science Committee, 1968.
- Olsen, G. **From COM to Common**. Queue, ACM Press, 2006, 4, 20-26.
- Szyperski, C. **Component Software: Beyond Object-Oriented Programming**. Addison-Wesley Longman Publishing Co., Inc., 2002.

André Santanchè

http://purl.org/andresantanche



License