Comunicação entre conteúdos e Learning Management Systems SCORM 2004

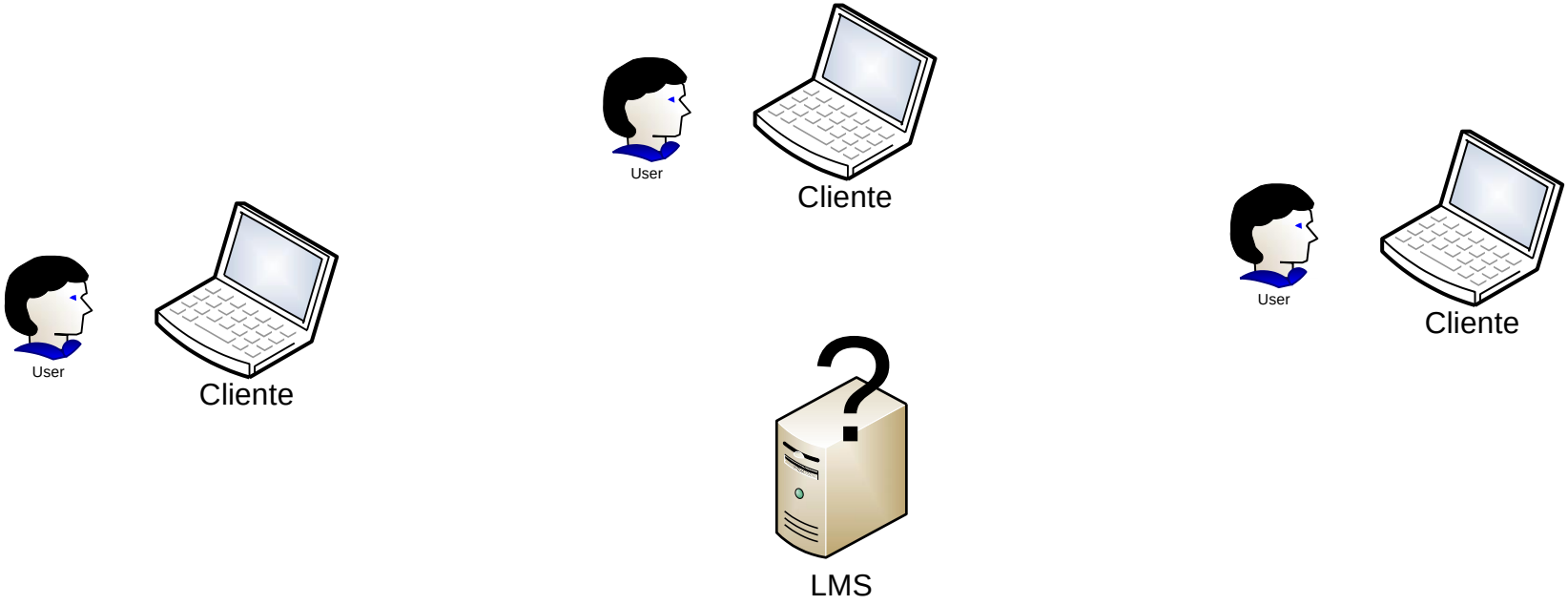
baseado no livro

"**SCORM 2004 Run-Time Environment [RTE]**"

disponível em:

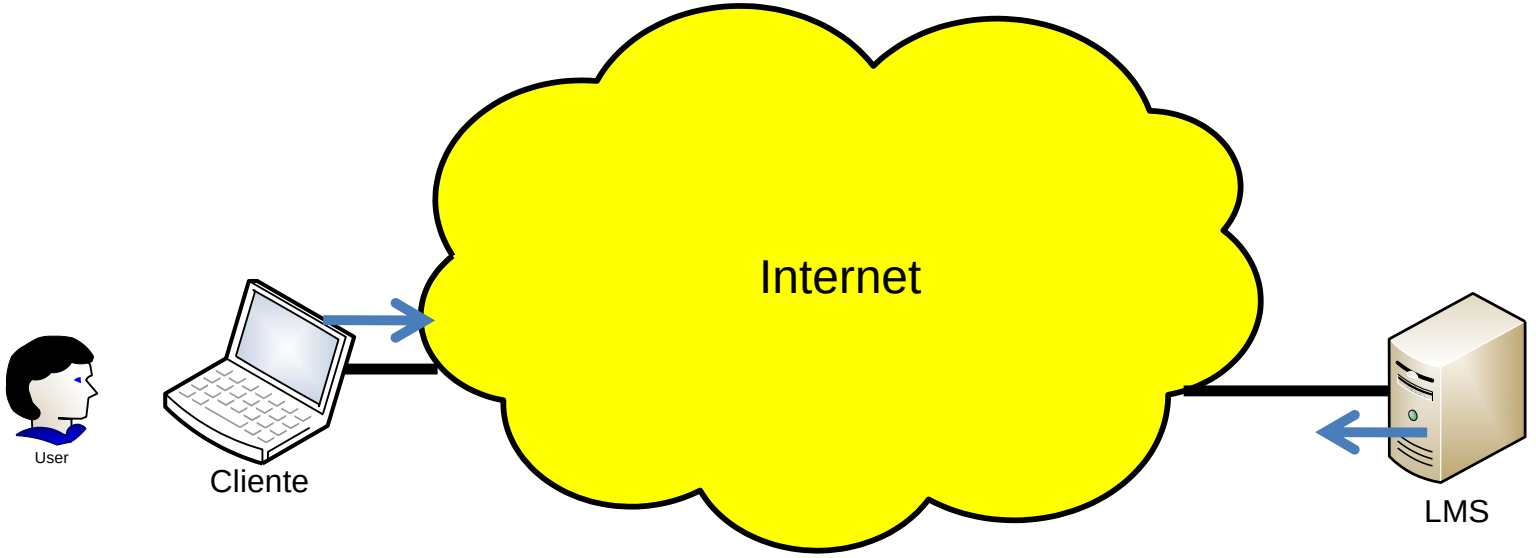
https://adlnet.gov/assets/uploads/SCORM_2004_4ED_v1_1_Doc_Suite.zip

Conceitos fundamentais



- O LMS necessita de obter informação sobre o processo de aprendizagem
  - Quando é que um objecto de aprendizagem começou a ser usado?
  - Que utilizador está a usar o objecto de aprendizagem?
  - Quando é que terminou a utilização desse objecto?
  - Num teste, que classificação foi obtida?
  - Que percentagem do objecto foi correctamente assimilada?
  - Porque razão é que o utilizador abandonou o objecto?

É necessária a troca de dados



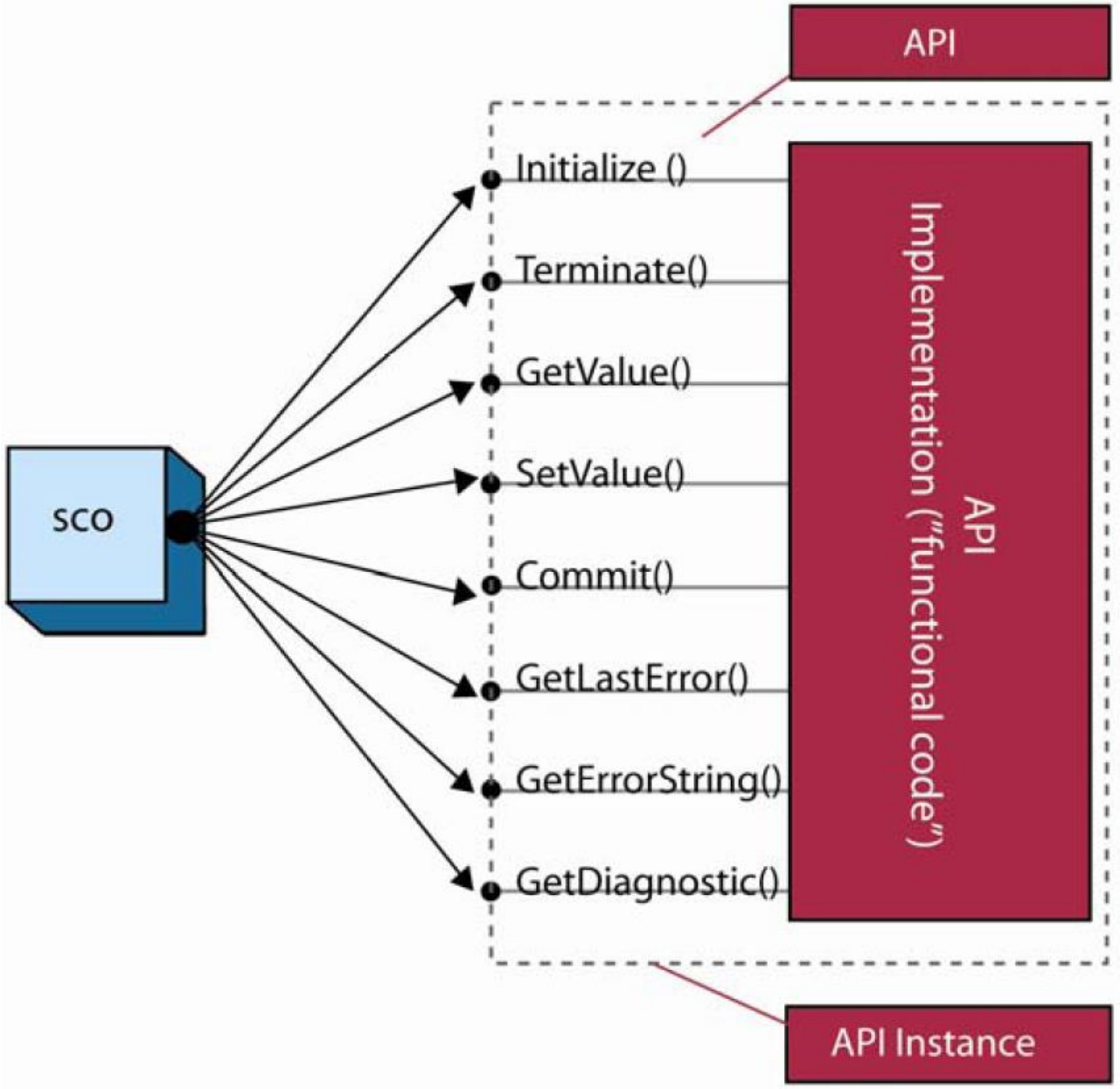
- Os conteúdos e o LMS têm que trocar dados
  - É necessária uma ligação contínua entre ambos;
  - Para garantir a interoperabilidade, é necessário um mecanismo normalizado de comunicação;
  - Para garantir a portabilidade dos conteúdos, estes não devem necessitar de saber à partida qual é o LMS onde vão ser usados (abstracção);

O modelo SCORM define…

- Normas para os ambientes “Run Time”;
- Uma API (Application Programming Interface) normalizada de comunicação;
- Um modelo de dados normalizado para usar na comunicação;

- No fundo, o modelo SCORM define a forma como os conteúdos e os LMS devem comunicar

As funções da API



Métodos da API

Sessão

- **doInitialize()** : indica ao LMS que a utilização do objecto começou
- **doTerminate()** : indica ao LMS que a utilização do objecto terminou

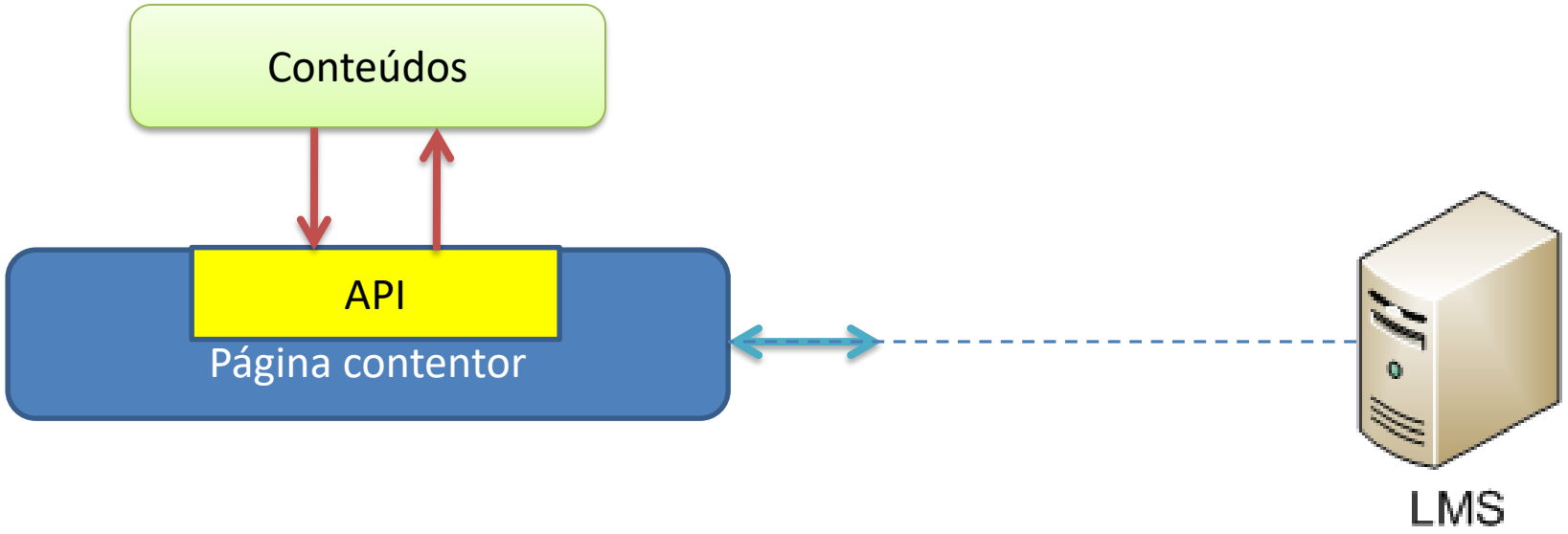
Dados

- **doGetValue(x)** : solicita ao LMS que forneça o valor do campo x
- **doSetValue(x,y)** : solicita ao LMS que atribua o valor y ao campo x
- **doCommit()** : envia para o LMS todos os dados eventualmente em cache

Suporte

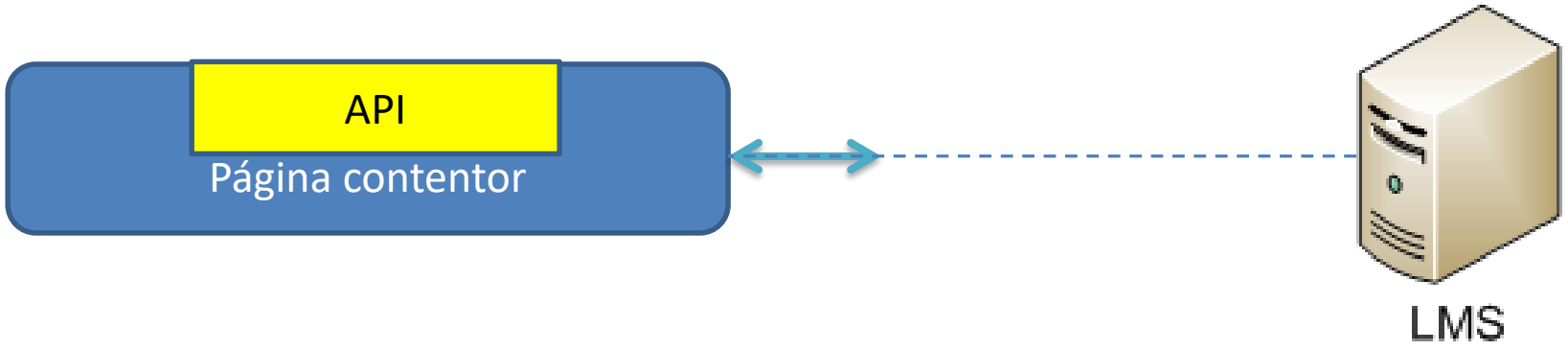
- **doGetLastError()** : solicita à API que indique o código do último erro que ocorreu
- **doGetErrorString(error_code)** : devolve a descrição de um código de erro
- **doGetDiagnostic(codigo)** : usada para obter informação adicional de diagnóstico

Como é feita a comunicação ?



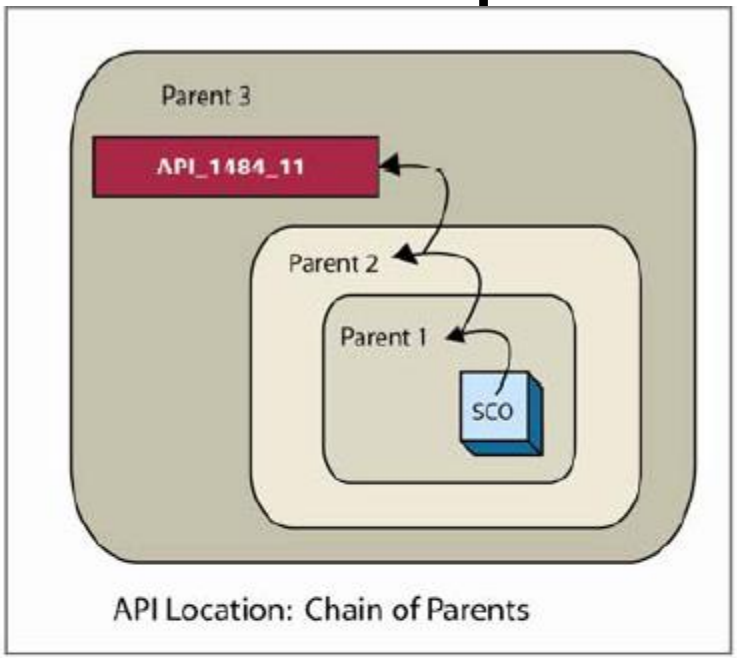
- Os conteúdos não correm directamente no browser, correm dentro de um contentor
- O contentor é específico de cada LMS e expõe a API aos conteúdos
- Os conteúdos não comunicam directamente com o LMS, comunicam com o contentor, através de Javascript, usando uma API normalizada
- O contentor comunica com o LMS usando um método específico de cada LMS

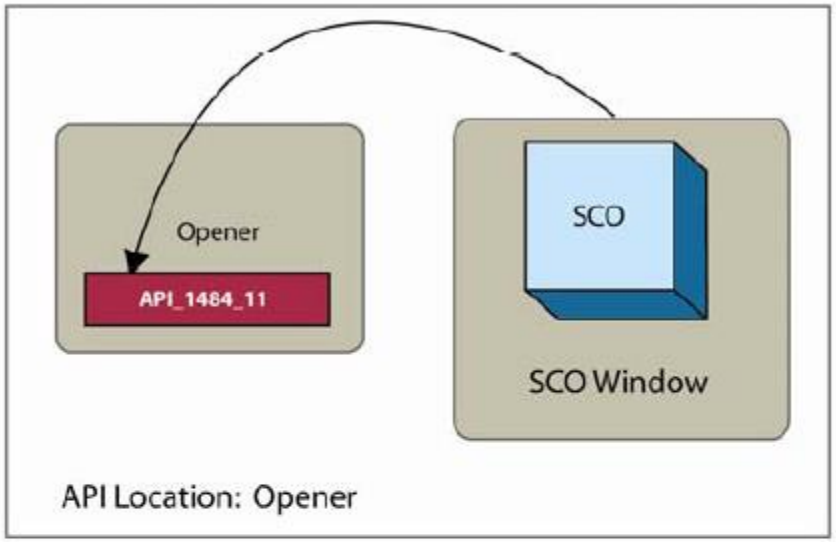
Responsabilidades do LMS

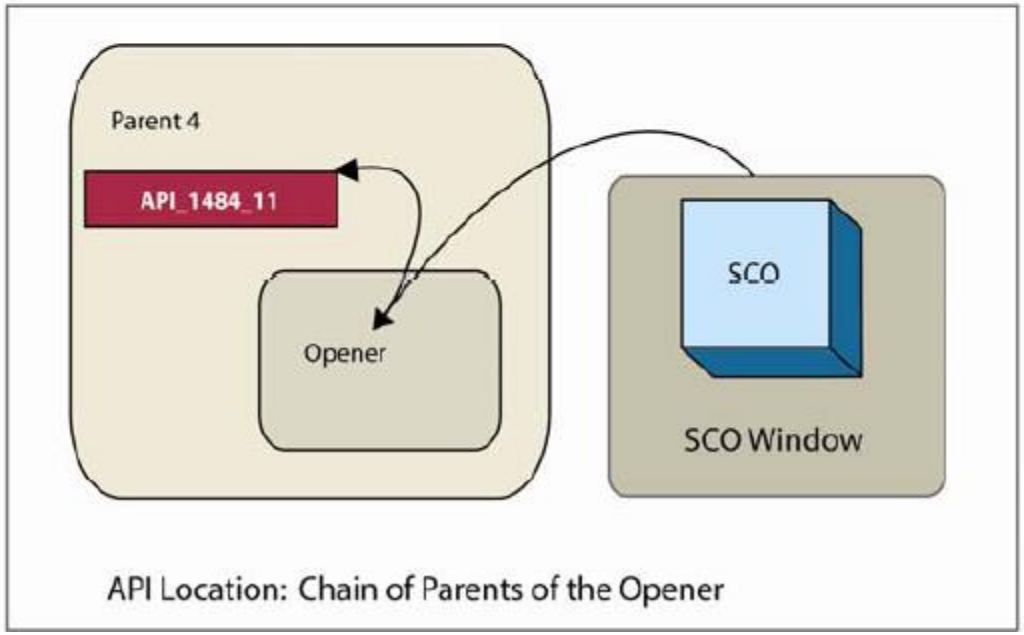


- O LMS tem que lançar os conteúdos dentro de uma hierarquia DOM particular, dentro de uma janela filha ou de uma frame filha da página que contém a API
- A página que contém a API deve actuar como um intermediário entre os conteúdos e o LMS

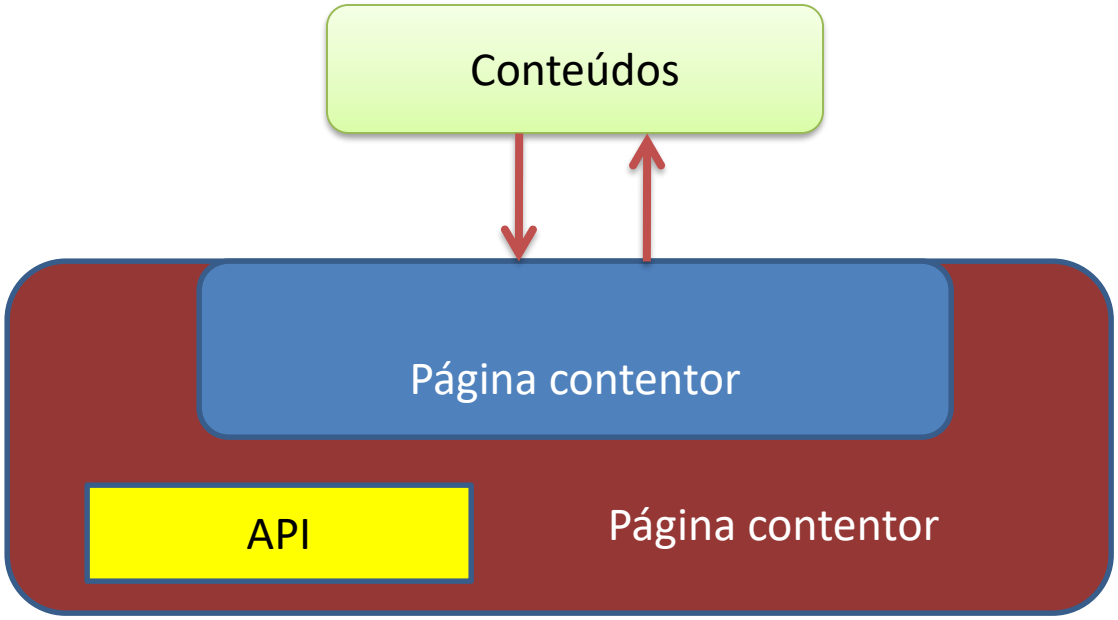
Hierarquias DOM permitidas







Responsabilidades dos conteúdos



- Os conteúdos devem começar por procurar a página que tem a API, percorrendo a hierarquia DOM em direcção ao topo, até a encontrar;
- Se encontrarem a API, devem usar obrigatoriamente os métodos:
  - doInitialize()
  - doTerminate()

Exemplo de código para procurar API

```
var nFindAPITries = 0;
var API = null;
var maxTries = 500;
var APIVersion = "";

function GetAPI(win) {
  if ((win.parent != null) && (win.parent != win)) {
    API = ScanForAPI(win.parent);
  }

  if ((API == null) && (win.opener != null)) {
    API = ScanForAPI(win.opener);
  }

  if (API != null) {
    APIVersion = API.version;
  }
}

function ScanForAPI(win) {
  while ((win.API_1484_11 == null) && (win.parent != null) && (win.parent != win)) {
    nFindAPITries++;
    if (nFindAPITries > maxTries) {
      return null;
    }
    win = win.parent;
  }
  return win.API_1484_11;
}
```

A utilização da API na prática

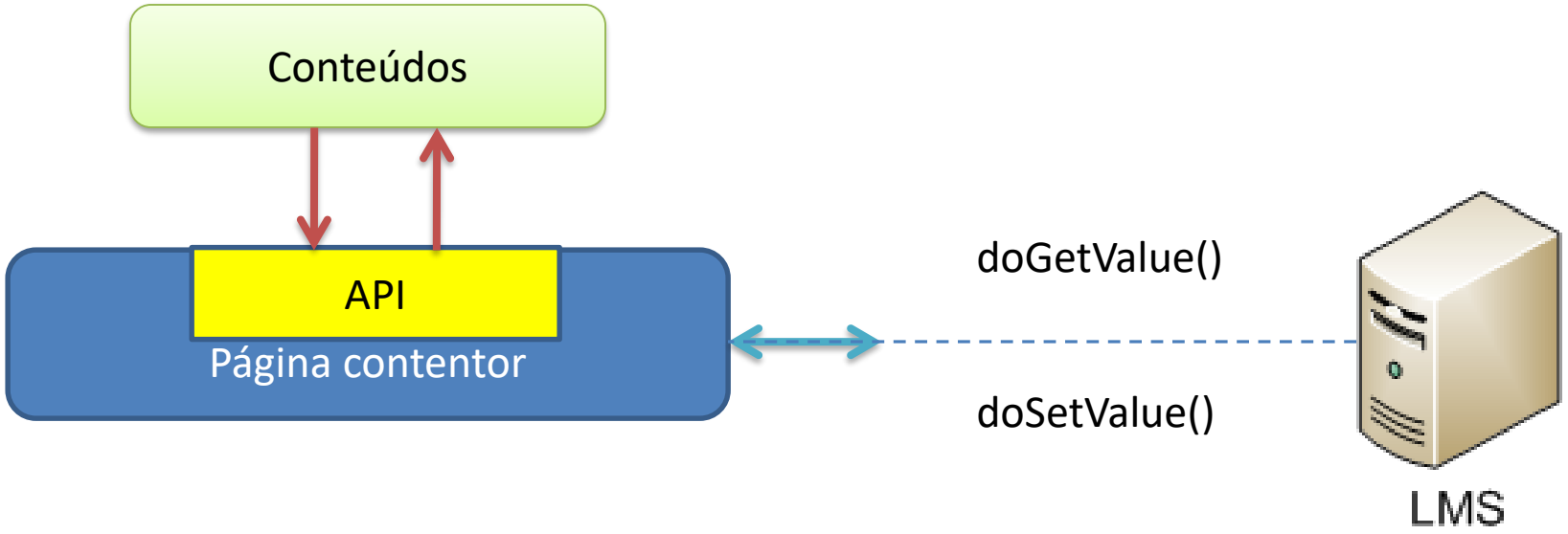
- É criado um módulo JavaScript comum (wrapper), que lida com a API
- Os objectos de aprendizagem partilham esse módulo
- O módulo contém funções que interagem com os métodos da API

Exemplo:

```
<html>
<head>
    <script type="text/javascript" src="APIWrapper.js"></script>
</head>

<body   onLoad="doInitialize()"   onUnload="doTerminate()"  >
```

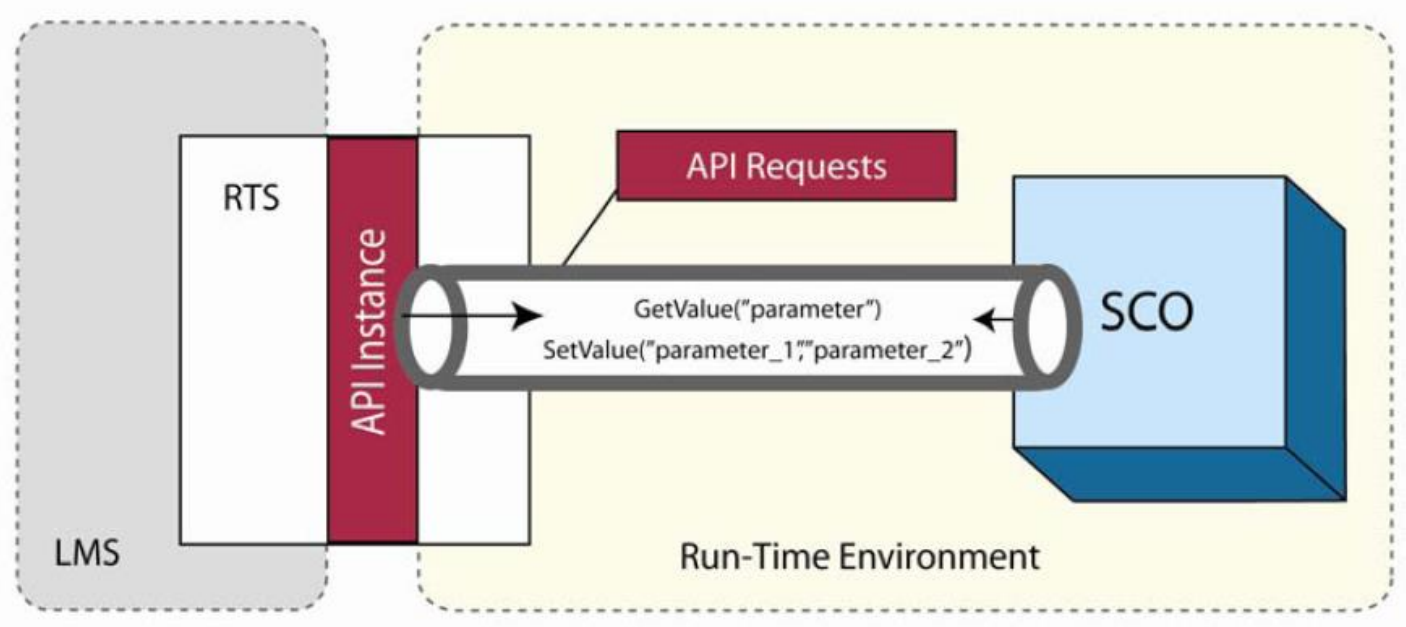
A API tem métodos para troca de dados



- O método doGetValue() permite aos conteúdos **obter** dados do LMS
- O método doSetValue() permite aos conteúdos **enviar** dados para o LMS

- Mas afinal… que dados são estes?
  - Estes dados estão definidos no Run Time Data Model

Objectivos fundamentais



- Definir um conjunto de dados normalizados que podem ser trocados entre os conteúdos e os LMS
- Esses dados, incluem:
  - Informações sobre o aluno;
  - Informações sobre o grau de sucesso ou de assimilação dos conteúdos;
  - Interacções que o aluno teve com os conteúdos;
- Estes dados podem ser usados pelo LMS para determinar o progresso do aluno, decidir sobre os próximos conteúdos a fornecer e gerar relatórios.

Elementos do modelo (1/2)

Data Model Element	Dot-Notation Binding	Description
Comments From Learner	cmi.comments_from_learner	Contains text from the learner.
Comments From LMS	cmi.comments_from_lms	Contains comments and annotations intended to be made available to the learner.
Completion Status	cmi.completion_status	Indicates whether the learner has completed the SCO.
Completion Threshold	cmi.completion_threshold	Identifies a value against which the measure of the progress the learner has made toward completing the SCO can be compared to determine whether the SCO should be considered completed.
Credit	cmi.credit	Indicates whether the learner will be credited for performance in this SCO.
Entry	cmi.entry	Contains information that asserts whether the learner has previously accessed the SCO.
Exit	cmi.exit	Indicates how or why the learner left the SCO.
Interactions	cmi.interactions	Defines information pertaining to an interaction for the purpose of measurement or assessment.
Launch Data	cmi.launch_data	Provides data specific to a SCO that the SCO can use for initialization.
Learner Id	cmi.learner_id	Identifies the learner on behalf of whom the SCO instance was launched.

Elementos do modelo (2/2)

Learner Name	cmi.learner_name	Represents the name of the learner.
Learner Preference	cmi.learner_preference	Specifies learner preferences associated with the learner's use of the SCO.
Location	cmi.location	Represents a location in the SCO.
Maximum Time Allowed	cmi.max_time_allowed	Indicates the amount of accumulated time the learner is allowed to use a SCO in the learner attempt.
Mode	cmi.mode	Identifies the modes in which the SCO may be presented to the learner.
Objectives	cmi.objectives	Specifies learning or performance objectives associated with a SCO.
Progress Measure	cmi.progress_measure	Identifies a measure of the progress the learner has made toward completing the SCO.
Scaled Passing Score	cmi.scaled_passing_score	Identifies the scaled passing score for a SCO.
Score	cmi.score	Identifies the learner's score for the SCO.
Session Time	cmi.session_time	Identifies the amount of time that the learner has spent in the current learner session for the SCO.
Success Status	cmi.success_status	Indicates whether the learner has mastered the SCO.
Suspend Data	cmi.suspend_data	Provides information that may be created by a SCO as a result of a learner accessing or interacting with the SCO.
Time Limit Action	cmi.time_limit_action	Indicates what the SCO should do when the maximum time allowed is exceeded.
Total Time	cmi.total_time	Identifies the sum of all of the learner's learner session times accumulated in the current learner attempt prior to the current

Exemplo de código

- Este exemplo mostra como:
  - Enviar o resultado de um teste;
  - Definir a escala de classificação;
  - Mudar o estado do objecto de aprendizagem

```
var pontuacaoTotal;
…….
//calcular pontuação total
…….
 //enviar pontuação
doSetValue("cmi.core.score.raw", pontuacaoTotal);
 //definir os limites mínimo e máximo
doSetValue("cmi.core.score.min", “0”);
doSetValue("cmi.core.score.max", “100”);

//mudar o estado do objecto
doSetValue("cmi.core.lesson_status", "completed");

//Enviar efectivamente todos os pedidos
doCommit();
```

Descrição detalhada dos elementos

A descrição detalhada dos elementos do modelo pode se consultada na secção 4 do livro "**SCORM 2004 Run-Time Environment [RTE]**"

disponível em:
https://adlnet.gov/assets/uploads/SCORM_2004_4ED_v1_1_Doc_Suite.zip