**黑龙江大学**

**实 验 报 告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **课程名称** | **计算机操作系统** | | | | | |
| **实验项目名称** | **设备管理** | | | | | |
| **实验时间**  **（日期及节次）** | **2016年10月14日 5-6节** | | | | | |
| **专业** | **计算机** | | **学生所在学院** | | **计算机科学技术学院** | |
| **年级** | **2014级** | | **学号** | | **20123349** | |
| **姓名** | **刘伟** | | **指导教师** | | **朱敬华** | |
| **实验室名称** |  | | | | | |
| **实验成绩** | **预习情况** | **操作技术** | **实验报告** | **附加：综合创新能力** | | **实验**  **综合成绩** |
|  |  |  |  | |  |
| **教师签字** |  | | | | | |

**黑龙江大学教务处**

**实验三 设备管理**

**一．实验名称：**设备管理

**二．实验目的：**

设备管理主要包括设备的分配和回收、同时实现设备独立性。

**三．实验内容：**

假定模拟系统中有A（3台）、B（2台）、C三种设备，采用安全分配方式。

1、设备管理子系统涉及到系统设备表（SDT）、通道控制表（CHCT）、控制器控制表（COCT）和设备控制表（DCT）来体现输入输出系统的四级结构和三级控制。我们模拟这样的数据结构来完成对外围设备的管理。

（1）添加设备：增加对应的设备控制表和系统设备表中的表项，如果需要新建对应的控制器控制表。

（2）删除设备：删除对应的设备控制表和系统设备表中的表项，如果需要删除对应的控制器控制表。

2、设备分配必须满足设备的独立性要求。为了实现设备独立性，要求在驱动程序之上设计一层设备无关软件，其主要功能可分为：

（1）执行所有设备的公有操作，主要包括：(a)独占设备的分配与回收；(b)将逻辑设备名映射为物理设备（LUT），进一步可以找到相应物理设备的驱动程序。

（2）向用户层（或文件层）软件提供统一的接口。例如，对各种设备的读操作，在应用程序中都用read; 而对各种设备的写操作，则都使用write。

**四．实验代码**

var express = require('express');

var router = express.Router();

var DevArr = [];

var ColArr = [];

var ChaArr = [];

DevArr = [

{

devName: 'Keyboard',

devType: 'I',

Status: '0',

devCol: 'CO1',

devQue:[]

},

{

devName: 'Mouse',

devType: 'I',

Status: '0',

devCol: 'CO1',

devQue:[]

},

{

devName: 'Display',

devType: 'O',

Status: '0',

devCol: 'CO2',

devQue:[]

},

{

devName: 'Printer',

devType: 'O',

Status: '0',

devCol: 'CO3',

devQue:[]

}

];

ColArr = [

{

colName: 'CO1',

Status: '0',

colCha: 'CH1',

colQue: []

},

{

colName: 'CO2',

Status: '0',

colCha: 'CH2',

colQue: []

},

{

colName: 'CO3',

Status: '0',

colCha: 'CH2',

colQue: []

}

];

ChaArr = [

{

chaName: 'CH1',

Status: '0',

chaQue: []

},

{

chaName: 'CH2',

Status: '0',

chaQue: []

}

];

var isCol = function(devCol,ColArr){

var bFlag = 0;

for(var i=0; i<ColArr.length; i++){

if(ColArr[i].colName == devCol){

bFlag = 1;

break;

}else{

bFlag = 0;

}

}

return bFlag;

};

var devIndex = function(devName){

for(var i=0; i<DevArr.length; i++){

if(DevArr[i].devName == devName){

return i;

}

}

};

var colIndex = function(colName){

for(var i=0; i<ColArr.length; i++){

if(ColArr[i].colName == colName){

return i;

}

}

};

var getDevObj = function(devName){

for(var i=0; i<DevArr.length; i++){

if(DevArr[i].devName == devName){

return DevArr[i];

}

}

};

var getColObj = function(devName){

var devObj = getDevObj(devName);

var colName = devObj.devCol;

for(var i=0; i<ColArr.length; i++){

if(ColArr[i].colName == colName){

return ColArr[i];

}

}

};

var getChaObj = function(devName){

var colObj = getColObj(devName);

var chaName = colObj.colCha;

for(var i=0; i<ChaArr.length; i++){

if(ChaArr[i].chaName == chaName){

return ChaArr[i];

}

}

};

var isBusy = function(itemObj){

var bFlag = 0;

bFlag = itemObj.Status == '1'?1:0;

return bFlag;

};

/\* GET home page. \*/

router.get('/', function(req, res) {

res.render('index',{

DevArr: DevArr,

ColArr: ColArr,

ChaArr: ChaArr

});

});

router.get('/checkCol', function(req, res) {

var devCol = req.query.devCol;

if(isCol(devCol,ColArr)){

res.send('success');

}else{

res.send('fail');

}

});

router.post('/addDev', function(req, res) {

var devName = req.body.devName;

var devType = req.body.devType;

var devCol = req.body.devCol;

if(isCol(devCol,ColArr)){

DevArr.push({

devName: devName,

devType: devType,

Status: '0',

devCol: devCol,

devQue: []

});

res.redirect(303,'/');

}else{

var colCha = req.body.colCha;

DevArr.push({

devName: devName,

devType: devType,

Status: '0',

devCol: devCol,

devQue: []

});

ColArr.push({

colName: devCol,

Status: '0',

colCha: colCha,

colQue: []

});

res.redirect(303,'/');

}

});

router.post('/delDev',function(req,res){

var devName = req.body.devName;

var colObj = getColObj(devName);

DevArr.splice(devIndex(devName),1);

ColArr.splice(colIndex(colObj.colName),1);

res.redirect(303,'/');

});

router.post('/appDev',function(req,res){

var procName = req.body.procName;

var devName = req.body.devName;

var devObj = getDevObj(devName);

var colObj = getColObj(devName);

var chaObj = getChaObj(devName);

if(isBusy(devObj)){

devObj.devQue.push(procName);

}else{

devObj.Status = '1';

if(isBusy(colObj)){

colObj.colQue.push(procName);

}else{

colObj.Status = '1';

if(isBusy(chaObj)){

chaObj.chaQue.push(procName);

}else{

chaObj.Status = '1';

devObj.devQue.push(procName);

}

}

}

res.redirect(303,'/');

});

router.post('/relDev',function(req,res){

var devName = req.body.devName;

var devObj = getDevObj(devName);

var colObj = getColObj(devName);

var chaObj = getChaObj(devName);

var devQueArr = devObj.devQue;

var colQueArr = colObj.colQue;

var chaQueArr = chaObj.chaQue;

if(devQueArr.length>0){

devQueArr.shift();

if(devQueArr.length == 0){

devObj.Status = '0';

colObj.Status = '0';

chaObj.Status = '0';

}

}

if(colQueArr.length>0){

colQueArr.shift();

if(colQueArr.length == 0){

colObj.Status = '0';

}

}

if(chaQueArr.length>0){

chaQueArr.shift();

if(chaQueArr.length == 0){

chaObj.Status = '0';

}

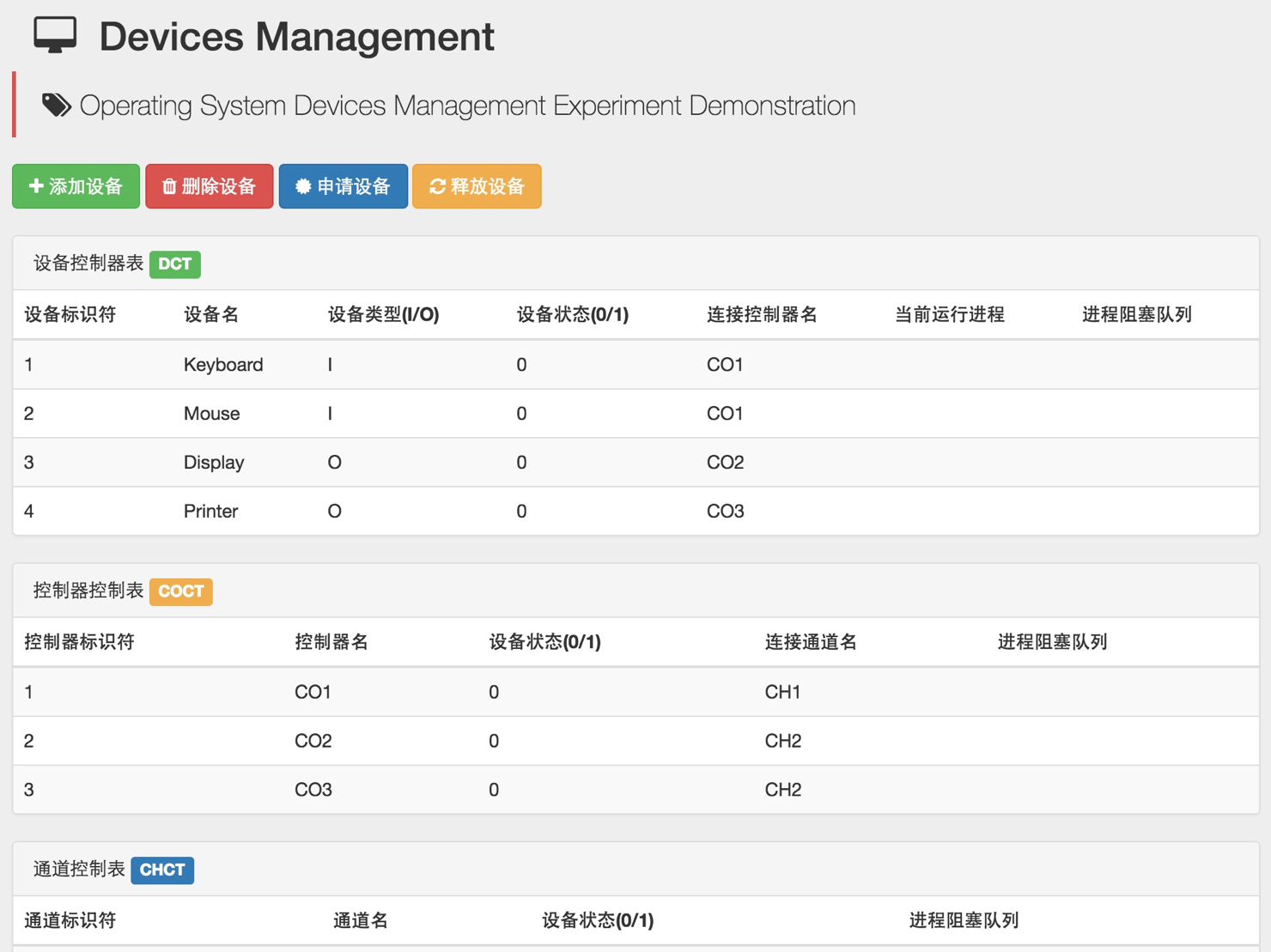
}

res.redirect(303,'/');

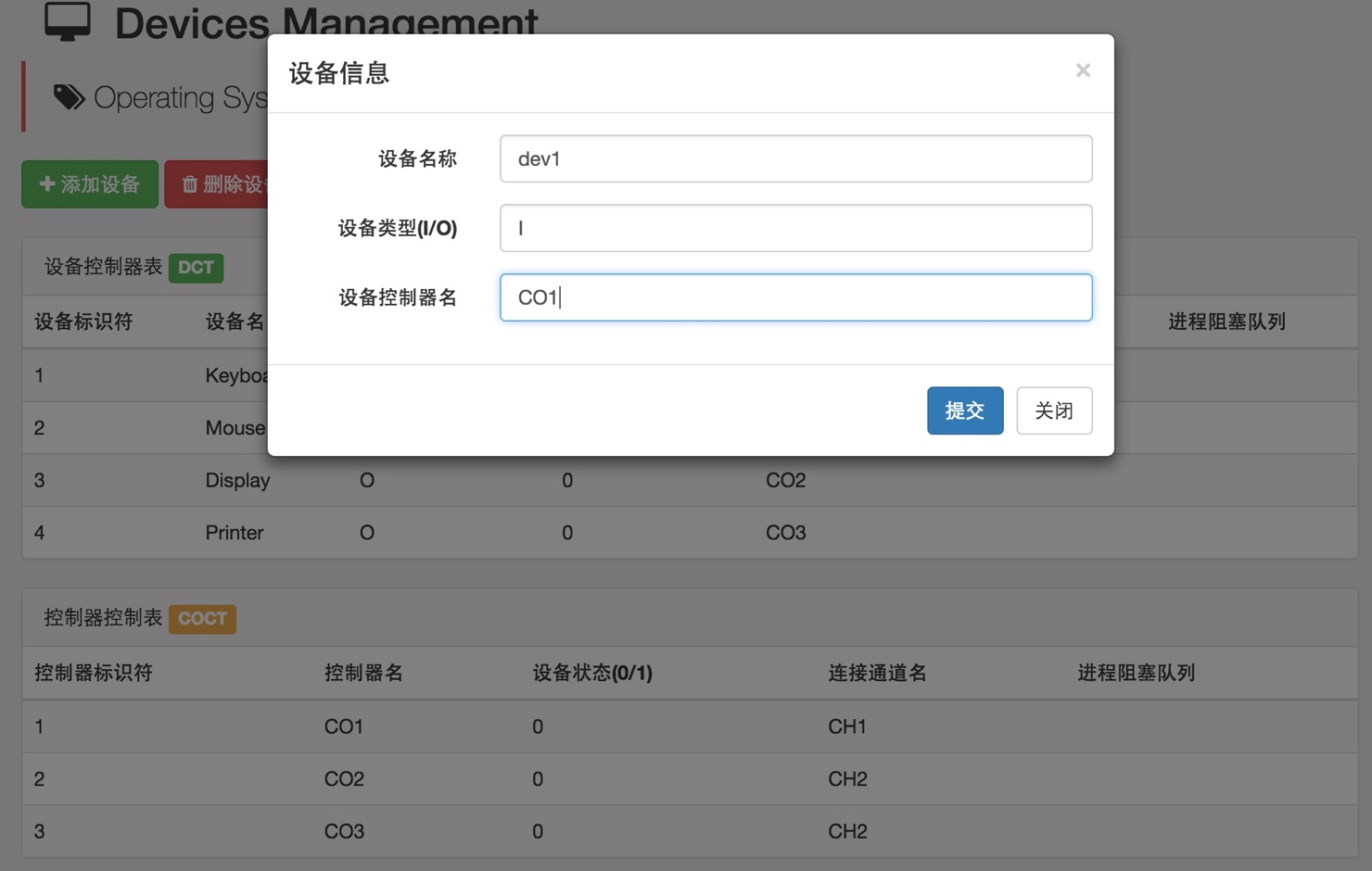
});

module.exports = router;

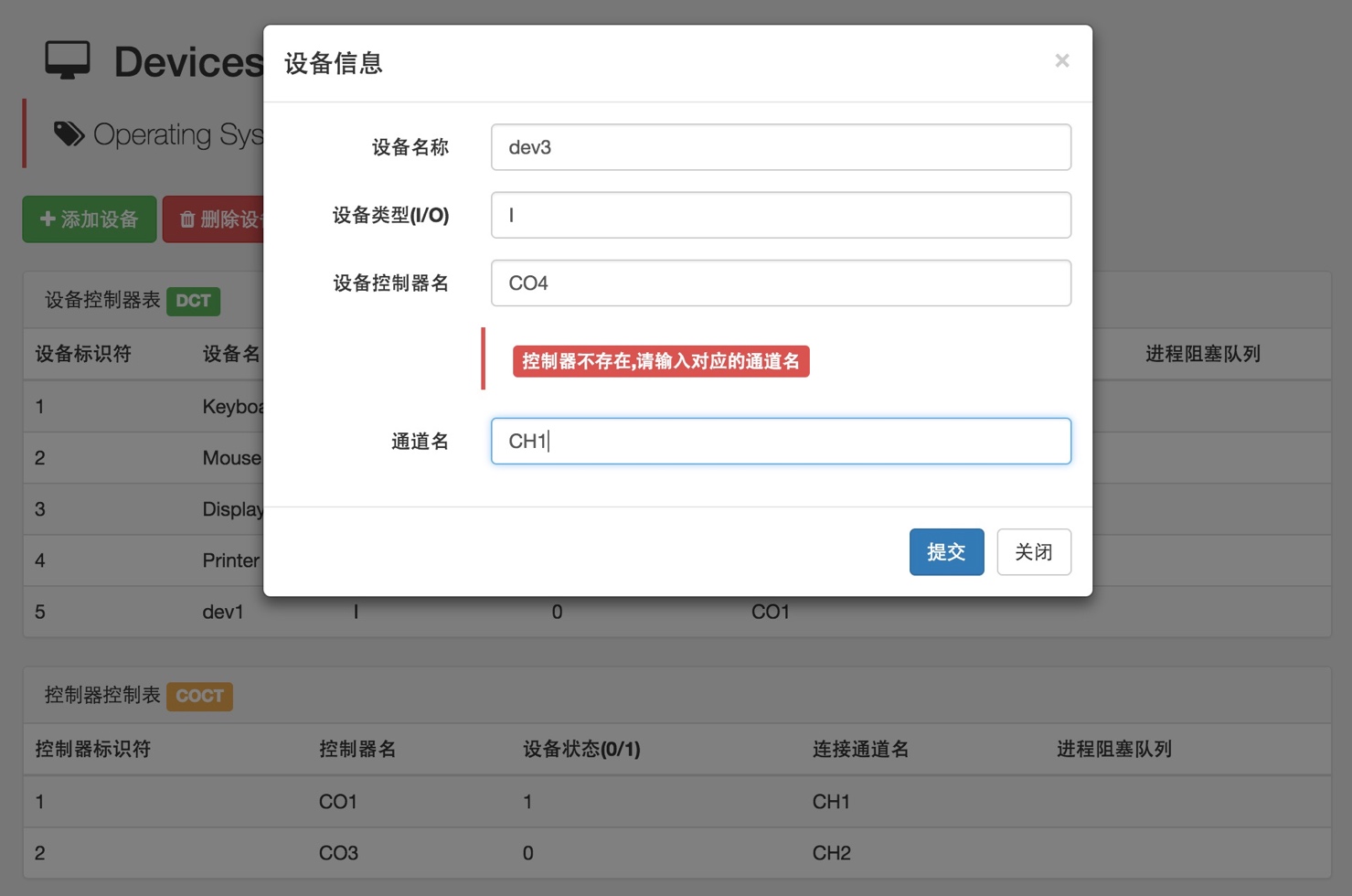
**五．实验截图：**

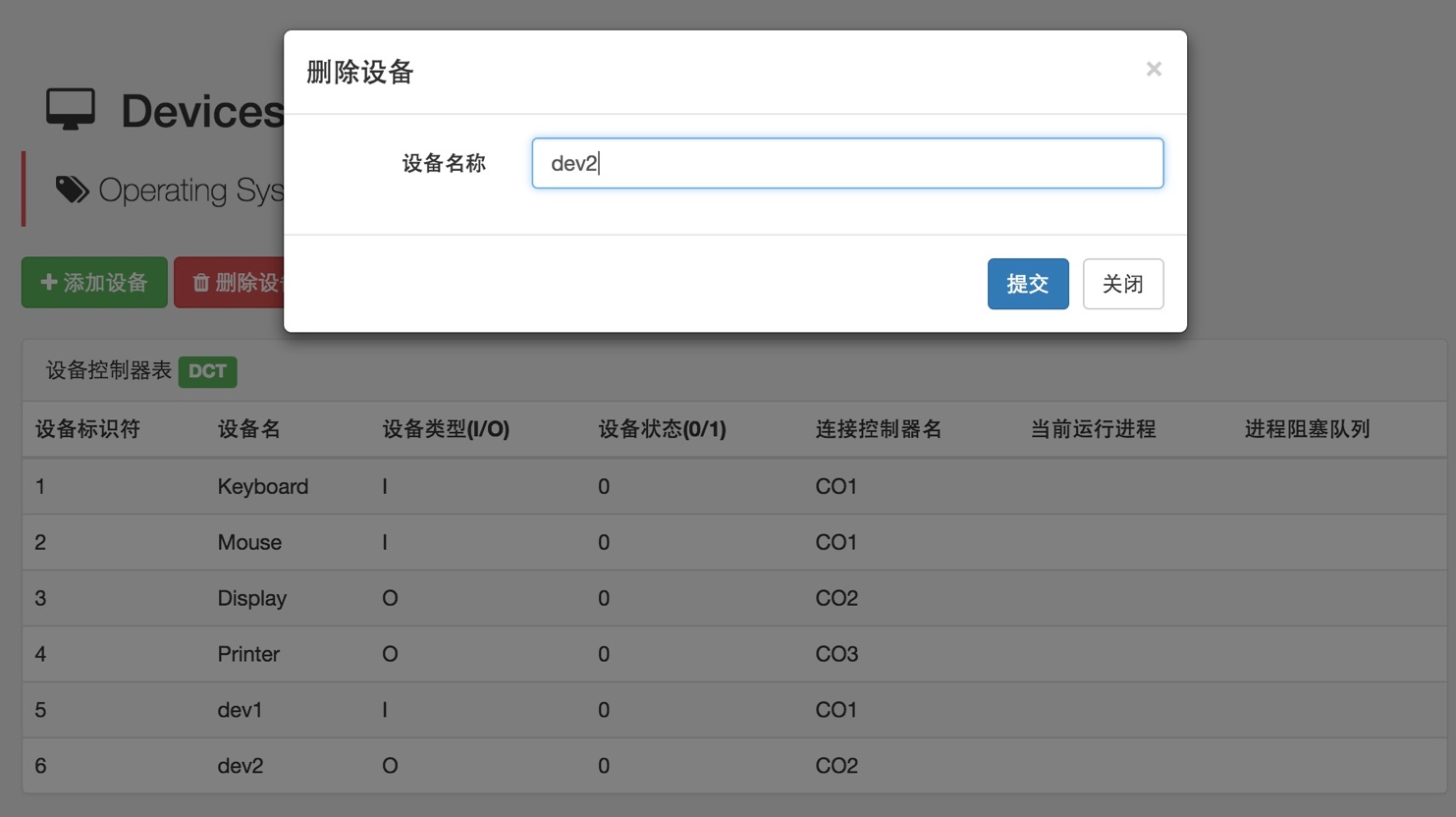
****



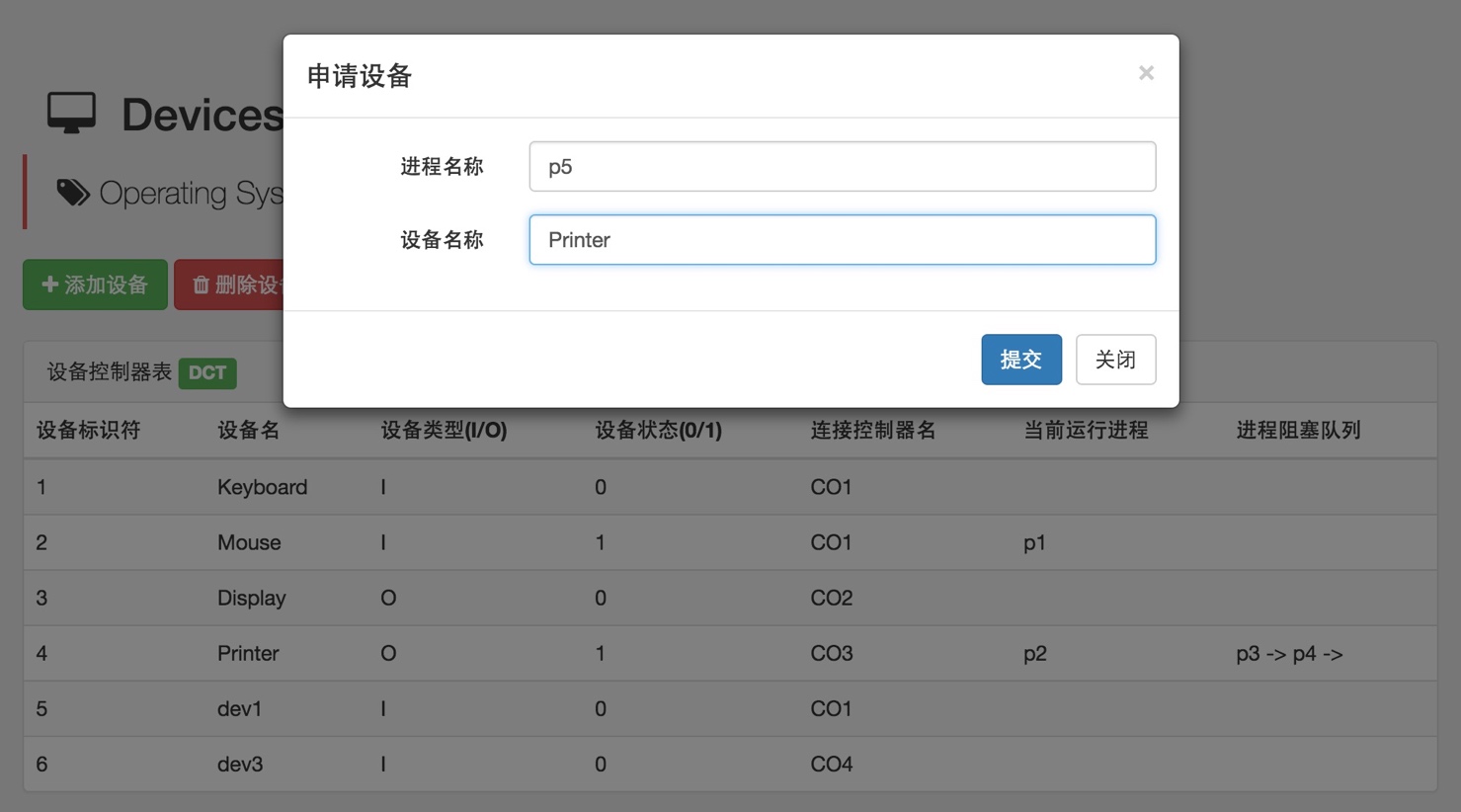














**六．实验心得：**

通过本次实验，了解了设备管理方式。了解了删除设备、添加设备的基本方式。同时了解了设备管理主要包括设备的分配和回收、同时实现设备独立性的基本方式和基本原理。