CHAPTER

# BI DESIGN AND DEVELOPMENT 14

## INFORMATION IN THIS CHAPTER:

- BI design
- BI user interface
- Privacy, security, access standards
- Design methods
- Prototyping lifecycle
- Application development tasks
- BI application testing

## BI DESIGN

The business intelligence (BI) content specifications document all the data processes in the BI application—access, integration, transformation, and analysis. The next step is to design the BI application's visual layout and how interacts with its users. This includes creating and adhering to standards for the user interface, and standards for how information is accessed from the perspectives of privacy and security.

Data visualization is a related topic because it also concerns the pictorial presentation information. As it is closely tied to analytics, it is covered in Chapter 15.

## BI USER INTERFACE (UI) STANDARDS

Websites are designed with UI standards that make the layout, navigation, and functionality consistent across all pages. This consistency makes us more productive; we can immediately use the Website for its intended purpose without having to figure out how it works. Likewise, it is critical for the BI team to implement UI standards for all BI applications so people using them can focus on what they need to accomplish.

In keeping with the Website analogy, the Web and BI development communities face similar challenges as depicted in Table 14.1.

Although they face similar challenges, there is a significant difference in results. The Web developer can create a Website that can be deployed on any Web browser, perform various functions, and be delivered on different platforms, yet still be consistent. But that consistency typically eludes BI developers.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Copyright © 2015 Elsevier Inc. All rights reserved.

Sherman, Rick. Business Intelligence Guidebook : From Data Integration to Analytics, Elsevier Science & Technology, 2014. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/felician-ebooks/detail.action?docID=1832704.
Created from felician-ebooks on 2022-10-18 14:53:40.

**Table 14.1 Web and Business Intelligence (BI) Development Challenges**

| Areas Affecting User Interface | Web | BI |
|---|---|---|
| Consumer interactions | Different Web browsers such as Chrome, Firefox, Internet Explorer, etc. | Different BI styles such as reports, dashboards, pivot tables, etc. |
| Application functions | Read, search, shop, social media interactions, etc. | Different types of analysis such as sales figures, future customer behavior, profitability, etc. |
| Delivery platforms | Web browsers, different-size monitors, tablets, smartphones, print, PDF, etc. | Same as Web |

The Web community's response to the various challenges it faced was to participate in an international standards consortium, World Wide Web Consortium, to develop UI standards covering customer-facing applications, application functionality, and delivery platforms. These standards include HTML (hypertext markup language), DHTML (dynamic HTML), cascading style sheets (CSS), extensible markup language (XML), asynchronous JavaScript and XML (AJAX), portable network graphics (PNG), scalable vector graphics (SVG), and many others. These standards, because product vendors adhere to them, ensure that the consumer has a consistent UI and the developers can concentrate on content rather than on all the differences in UI across the platforms, functions, and customer interactions.

On the other hand, the BI community still faces significant differences between vendors' products and BI styles. Although they follow a few standards, such as the database language and access protocols along with the Web standards just mentioned, those were agreed to by communities outside of BI. When it comes to customer interactions and application functions, rather than collaborating on industry standards, the BI product vendors continue to either view the differences as their edge over the competition or see standards as actually being harmful by commoditizing their products. This has resulted in differences between the various BI styles, how various product vendors implement the same BI style, and how the various analytical functions or charts are implemented. Because many BI product suites were created through product acquisitions, there is even an inconsistency in UI between the different tools within the suites.

The competition in the BI market has prevented the type of collaboration that is seen in the Web community, placing a burden on BI project teams to develop a group of UI standards to ensure a consistent UI experience for their customers. The BI UI standards need to cover the following areas:

- **Layout standards specific for each BI style** such as a report, dashboard, scorecard, or pivot table that is planned to be used. Each of these BI styles has unique layout and functional characteristics that need to be standardized every time they are used within the enterprise's BI applications. Examples of standardized that need to be addressed include:
  - Dashboards and scorecards—the minimum and maximum number of panels that can be used, the sizing and orientation of those panels, to the position of filters in, and the workflow between panels and other dashboards.
  - Pivot tables—how dimensions and measures are displayed and selected
  - Reports—whether production reports be pixel-perfect and static, what interaction people will have with the reports, how they will be delivered, and if people will be able to print or save reports as specific formats.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

- **Common layout standards** need to be defined across all BI styles and delivery platform.
  - Layout—backgrounds, BI applications' branding, and logos.
  - Text—fonts and styles for title, subtitles, labels, etc.
  - Color—color scheme for backgrounds, panels, labels, and chart types.
  - Filters, parameters, and slicers—how they are displayed, function, and are related to each other.
- **Delivery platform standards** need to be defined for each medium in which the BI application will be used (e.g., Web browser, notebook, tablet, smartphone, print, PDF). This also includes determining how to handle the differences in wide-screen monitors and notebooks, each of which is being viewed on a Web browser but with size differences drastically affecting what a person can see and interact with on the screen.
- **Chart type standards** need to be defined to match each type of analysis with the chart types that best supports it.

Many UI standards need to be developed for BI applications. It's a good idea to develop these standards on an as-needed basis rather than dedicating a significant a front portion of the project to defining them. Developing UI standards on an as-needed basis has these advantages:

- Supports an iterative and incremental approach, as advocated in Chapter 18, which in the long run will be faster than a waterfall methodology where all of the standards are defined up front, delaying when BI application deliverables reach the business people.
- Defining the UI standards when they are needed provides the context of immediately available use cases.

See the book's Website www.BIguidebook.com for a UI template.

Although some BI teams will create UI standards solely using text, it's best to supplement text with one of the following visual techniques: sketches, wireframes, storyboards, or mockups to better depict what the UI will look like. The old saying a picture is worth 1000 words is absolutely correct in this instance. And if time and resources permit, use a proof of concept, pilot, or prototype to implement the proposed UI standards in selected BI applications to obtain both developer and user feedback to better refine and validate standards.

The UI standards need to be built into templates for each BI tool, BI style, and delivery platform that will be used. Templates ensure consistency and improve productivity for both the business people and developers.

## CREATE PRIVACY, SECURITY AND ACCESS STANDARDS

Although privacy, security, and access are often intermixed, it is import to examine and establish standards for each.

First, establish standards for data that should not be disseminated because of regulations or a code of conduct. This can include data on customers, prospects, suppliers, employees, and more. In this age of Big Data and the Internet of Things, where the collection and analysis of data about people, places,

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

and things is rapidly increasing, a BI team must work with the business to define standards and enforce them. This enforcement will extend to both business and IT.

Second, create security standards on how to determine what data needs to be secured and how that will be implemented in the BI applications. The implementation may involve establishing a data schema with the name and pertinent attributes to resolve security access. Enterprise security regarding the BI applications and the data that they access will involve business and IT staff beyond the immediate BI team and stakeholders. Networking, server, storage, database, and file access are all part of the security framework. BI security standards establish what the BI applications need to do to contribute to the overall enterprise security.

Finally, with privacy and security standards in place, a BI team can work with the business to define access standards. These standards are not about declining someone's access for security or privacy reasons, but rather because it just makes them more productive. Typically, access standards specify BI consumers' attributes that will determine their access right. For example, someone's business group combined with their title, role, and location forms a set of attributes that many enterprises use to define access rights. The BI application using these access standards may expose data along product lines or geographies based on these attributes.

## DESIGNING EACH BI APPLICATION

Using the BI content specifications and UI standards, the BI designers should design the visual layout and workflow of each BI application. The deliverable of this design task is either a written specification or working prototype.

Historically, when the BI style was exclusively a printed report, the written specification would be straightforward and likely entirely in text. They would include a report header and footer, and a table grid in the report body. Since that time, many different BI styles have emerged that support a wide variety of visualizations. Dashboards started the trend of a BI application being a mashup of other BI applications. For example, a dashboard will contain multiple charts, each of which could be considered its own BI application. In addition, the mashup of BI applications will contain a workflow between them and may be interrelated, sharing filters, and drill-down or drill-across paths.

Just as with developing UI standards, using visual design methods improves the understanding of what is being proposed and encourages business people to provide feedback. The various techniques that BI designers use in creating BI application specifications are depicted in Figure 14.1. The methods are displayed in the order from left to right of increasing visual detail. A guiding principle for selecting and using one or more of these visual design methods is that they should be done quickly and not become projects unto themselves. The deliverables from these methods are design needs, not masterpieces of art.

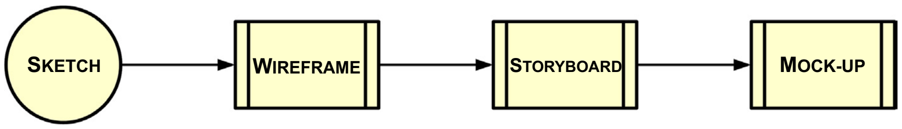


FIGURE 14.1

Business intelligence (BI) visual design methods.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

A working prototype may be used as a follow-on to any of the visual design methods or to replace these methods altogether. To develop a working prototype the following conditions must be met:

- Target schema with sample data must be available
- BI tool must be installed
- Prototyping environment must be available
- Team must have some level of development experience with a BI tool

We will discuss these prototypes for design and development later in this chapter.

### *Sketches*

Most visual designs will start off as a sketch as depicted in Figure 14.2. Using a whiteboard or paper is a quick, easy, and inexpensive way to draw the initial sketches and begin discussions. A drawback of using a whiteboard or paper is that it is difficult to get it into the specification document and share with others. With that in mind, there are drawing applications that can save the output into various formats, and collaborative applications that enable sharing by providing a virtual whiteboard. Another benefit of these applications is that you can use prebuilt shapes, making the sketches faster to draw, but more importantly, much neater. (From my diagram, you can see I need help drawing straight lines!)

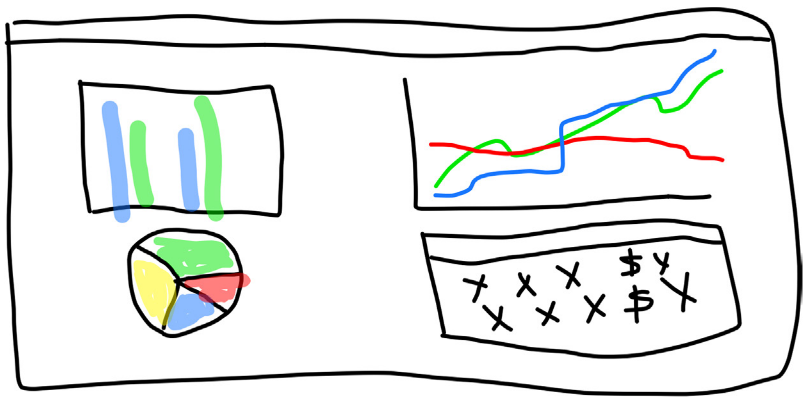


**FIGURE 14.2**

Whiteboard sketch.

Sketches enable brainstorming and participation since the process of creating them is quick, fluid, and disposable. Even though sketches are minimally detailed, they visually portray the structure of the BI application that text alone would not. Sketches are an excellent method to discuss and communicate the intent and content of the BI application from a conceptual view with the BI stakeholders.

Sketches are a starting point for the visual design of BI applications, but they are still ambiguous and you need to add more detail using other visual design techniques.

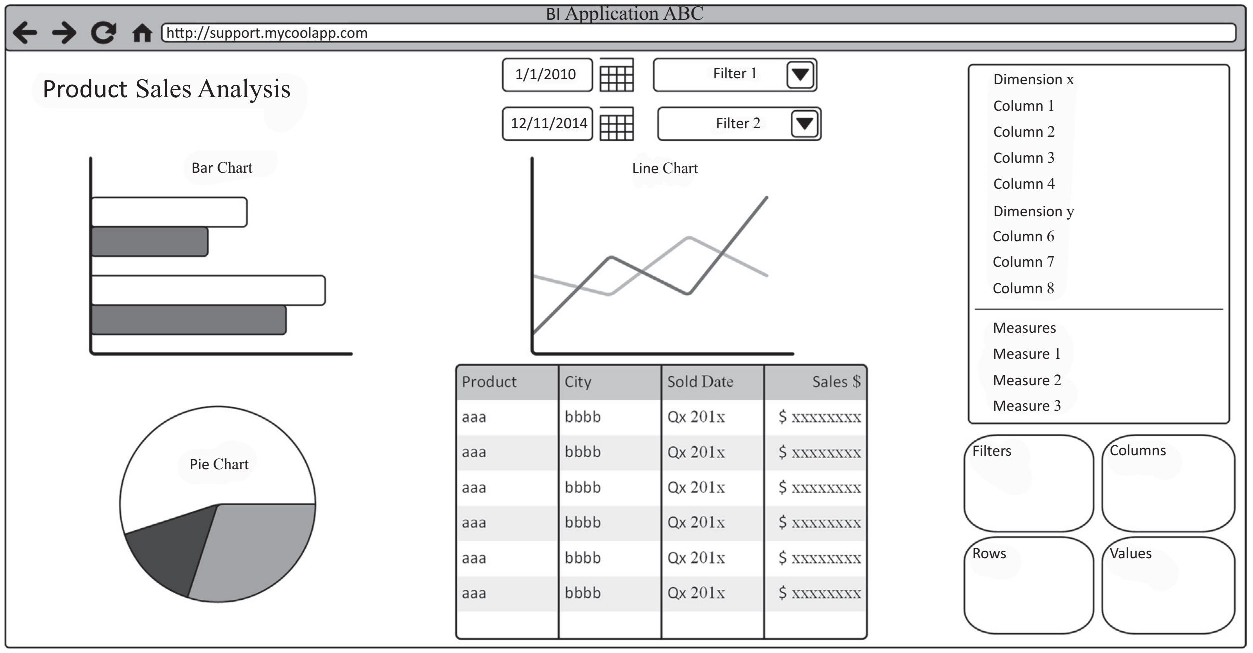
### *Wireframes*

Designing applications using wireframes is a technique that is used extensively in Web design. A wireframe, as depicted in Figure 14.3, is a much more detailed representation of the visual design than a sketch; however, the technique deliberately does not attempt to provide detailed visual design.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Sherman, Rick. Business Intelligence Guidebook : From Data Integration to Analytics, Elsevier Science & Technology, 2014. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/felician-ebooks/detail.action?docID=1832704.
Created from felician-ebooks on 2022-10-18 14:53:40.



| Product | City | Sold Date | Sales $ |
|---|---|---|---|
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |

**FIGURE 14.3**

Business intelligence (BI) application wireframe.

A wireframe is referred to as a low fidelity representation of the design. A wireframe is similar to a blueprint (low fidelity) of a house rather than a three-dimensional CAD/CAM picture (high fidelity). A convention for wireframes is that they are drawn in black, shades of gray, and white rather than in color. This convention is used to constrain the scope of the wireframe and focus its purpose.

A wireframe portrays the visual structure of the BI application and provides the next level of detail beyond sketches.

The wireframe drawing contains:

- Each data content group.
- Content location in a layout.
- A representation of the visualizations as a chart or table. Some wireframe tools have prebuilt visualizations such as bar, line, and pie charts, but if your tools do not or the particular visualization that will be used is not a prebuilt icon, then use a box as a placeholder.
- Description and placement of filters, sliders, titles, subtitle, text fields, logos, and any other layout item. Placeholder boxes are typical in those cases because this is a low fidelity design, and you want to do it quickly.

There are many inexpensive and easy-to-use wireframe tools. In addition, spreadsheets and presentation applications are commonly used because they are available and people already know how to use them. The wireframe design process should be quick and inexpensive, without requiring sophisticated technical abilities.

A wireframe focuses on how the BI application screen layout will look rather than the navigation or the functionality involved, which is addressed by mock-ups, storyboards, and prototypes.

### Storyboards

The storyboarding technique is used extensively in the film industry to design both live-action and animated movies. Sketches are created to represent scenes in the movie; they are then laid out in scene sequence, illustrating all the major changes in action. Similarly, the BI storyboard lays out all the major actions that occur when business people are performing analysis in the BI application. The scenes that are used in the BI storyboard are each analytical process the business person performs in the BI application.

BI storyboards focus on the business user interaction of the analytical processes within the BI application. Storyboarding recognizes how business people actually work with BI applications such as a dashboard in order to gain insight into the business. Business people will typically perform several analyses, some of which are dependent on each other, to determine if they need to perform some action and what that action is. The storyboarding technique enables the BI designer gain a better perspective on how each type of analysis is related and enables the development of a more effective layout.

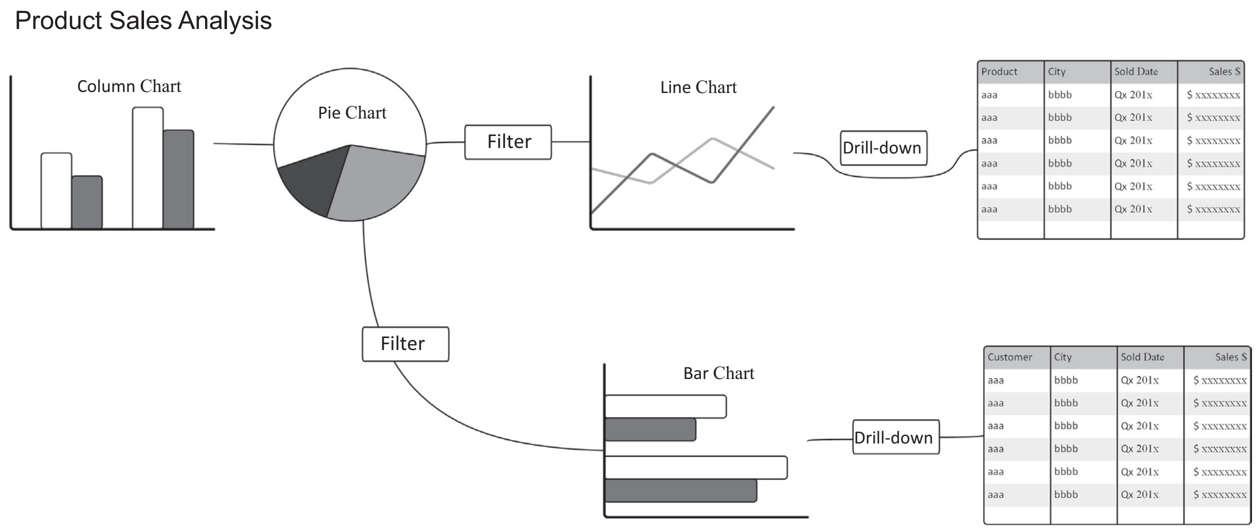
In the BI application depicted in Figure 14.3, the analytical processes or scenes are the comparison analysis (column chart), contribution analysis (pie chart), trend analysis (line chart), and detailed drill-down of the trend analysis (tabular grid). Figure 14.4 illustrates the workflow between each of these analysis processes laid out in sequence.

Storyboards are the most complete representation of visualization and workflow that a BI team can develop without constructing a working prototype.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Sherman, Rick. Business Intelligence Guidebook : From Data Integration to Analytics, Elsevier Science & Technology, 2014. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/felician-ebooks/detail.action?docID=1832704.
Created from felician-ebooks on 2022-10-18 14:53:40.



| Product | City | Sold Date | Sales $ |
|---|---|---|---|
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |

| Customer | City | Sold Date | Sales $ |
|---|---|---|---|
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |
| aaa | bbbb | Qx 201x | $ xxxxxxxx |

**FIGURE 14.4**

Business intelligence (BI) storyboard example.

### *Mock-ups*

Mock-ups are the medium- to high-fidelity representations of the BI application's layout. Returning to our analogy of the architectural drawings, the mockup is the three-dimensional CAD/CAM drawing of the house. The mockup is a static visualization of what the BI application will look like to the business person, including the colors, icons, buttons, text, and charts that will be used. It is a static representation so that business people cannot actually interact with it, but will be able to see what they will work with once it has been developed. Figure 14.5 depicts a mockup of the wireframe in Figure 14.3.

Mockups are most often used in creating multimedia items that are very expensive to create and produce. In those cases, they save time and money by working out the visual details beforehand. This is not a typical situation in most BI projects because the BI tools can create a prototype of the actual BI application's layout much more quickly than a mockup tool. In addition, the BI prototype can then be used to template. For these reasons, mockups are typically not used unless the BI tool creates them along with the prototype.

## BI DEVELOPMENT

Chapter 18 recommends a hybrid iterative and incremental project methodology. This hybrid methodology uses the traditional waterfall approach to design the overall BI framework and architecture while using an agile approach to develop the architecture's underlying components, such as the BI applications. Although I advocate an agile approach, this book is agnostic as to whether a formal agile project methodology needs to be used by the BI project manager. There are other alternatives that can be used to achieve the same results, as discussed in Chapter 18. From a development perspective, the BI team needs to build the BI applications incrementally and iteratively by using prototyping (described later).

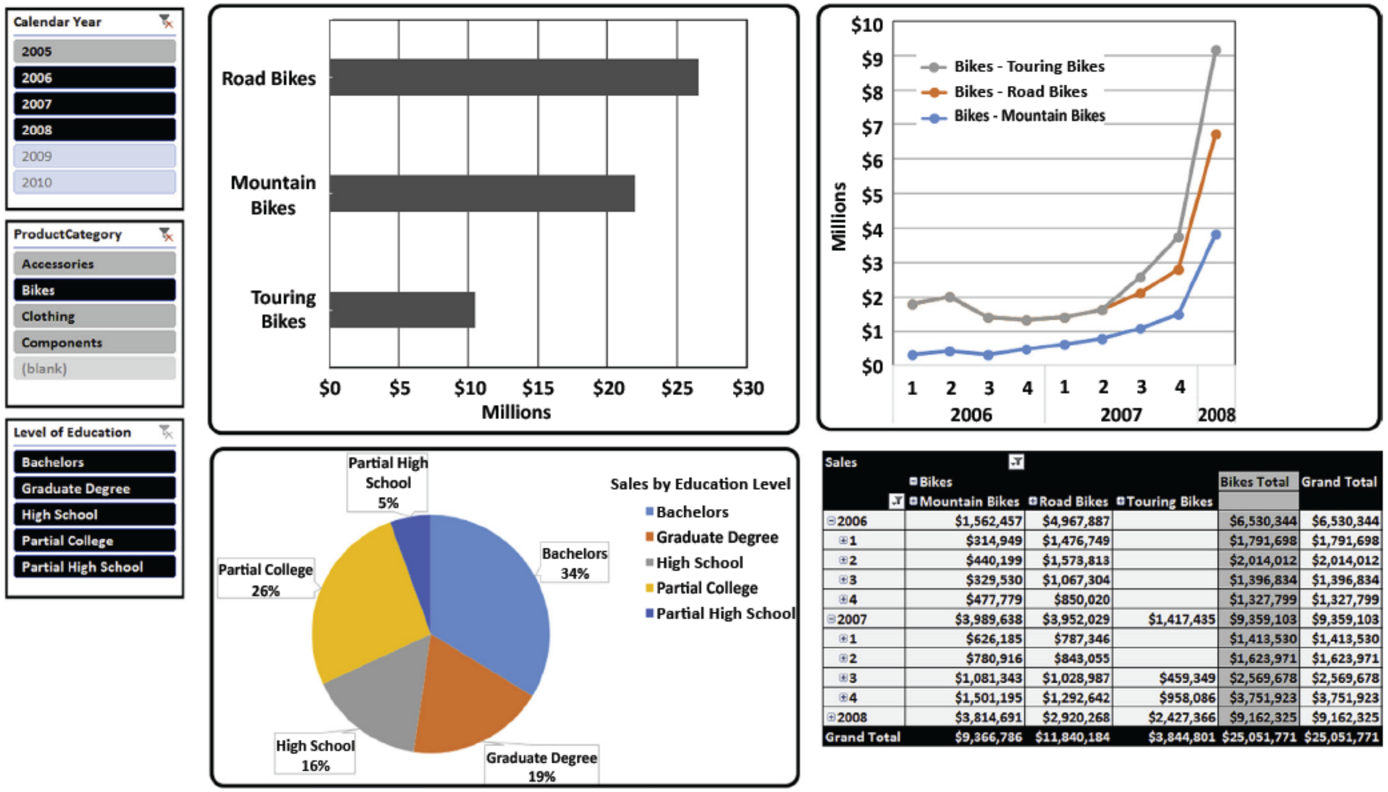
Note that this book uses the term *BI application* to mean the application that business people interact with to perform their reporting and analysis. The BI application may be built with any BI style such as reports, dashboards, scorecards, online analytical processing (OLAP) cubes, data discovery, predictive analytics, or advanced data visualizations. The assumption is that the data that the BI application uses is already available for the development team, but if not, then those data integration tasks discussed in the data integration section of this book will need to be done.

Business people are always getting surprised when BI applications are implemented. They will say that the application doesn't do what they asked for, is providing results different than expected or they need to add something they forgot to ask for. This scenario plays out time and time again. It is exacerbated when teams use a waterfall project methodology in which business people are engaged to gather requirements in the early stages of a project. Then, weeks or months later, when business people are reengaged to test the completed BI application, they notice different things. Although it's still a good idea to do the requirements early in the project to define scope, ensure that the overall framework is built, and to get the data in place for BI application development, switch to prototyping mode when you get to the BI application development phase. This gets people actively involved in a feedback loop and avoids the implementation surprises that plague so many BI projects.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Sherman, Rick. Business Intelligence Guidebook : From Data Integration to Analytics, Elsevier Science & Technology, 2014. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/felician-ebooks/detail.action?docID=1832704.
Created from felician-ebooks on 2022-10-18 14:53:40.



| Sales | Bikes | | | Bikes Total | Grand Total |
|---|---|---|---|---|---|
| | Mountain Bikes | Road Bikes | Touring Bikes | | |
| 2006 | $1,562,457 | $4,967,887 | | $6,530,344 | $6,530,344 |
| 1 | $314,949 | $1,476,749 | | $1,791,698 | $1,791,698 |
| 2 | $440,199 | $1,573,813 | | $2,014,012 | $2,014,012 |
| 3 | $329,530 | $1,067,304 | | $1,396,834 | $1,396,834 |
| 4 | $477,779 | $850,020 | | $1,327,799 | $1,327,799 |
| 2007 | $3,989,638 | $3,952,029 | $1,417,435 | $9,359,103 | $9,359,103 |
| 1 | $626,185 | $787,346 | | $1,413,530 | $1,413,530 |
| 2 | $780,916 | $843,055 | | $1,623,971 | $1,623,971 |
| 3 | $1,081,343 | $1,028,987 | $459,349 | $2,569,678 | $2,569,678 |
| 4 | $1,501,195 | $1,292,642 | $958,086 | $3,751,923 | $3,751,923 |
| 2008 | $3,814,691 | $2,920,268 | $2,427,366 | $9,162,325 | $9,162,325 |
| Grand Total | $9,366,786 | $11,840,184 | $3,844,801 | $25,051,771 | $25,051,771 |

**FIGURE 14.5**

Business intelligence (BI) mockup example.

## PROTOTYPING LIFECYCLE

Software prototyping is a technique to build specific portions of an application to investigate certain aspects such as functionality, feasibility, efficiency, performance, or how well it meets requirements. There are two primary objectives for BI prototypes:

- Obtain feedback from its intended users.
- Enable developers to build BI applications in an incremental manner.

There are several critical success factors in the BI prototyping efforts:

- **Dedicate time and effort**. Business people who will be using the BI applications in their jobs need to be actively involved in working with the BI prototypes and providing feedback to the BI developers. Business management needs to make a commitment that these people will have dedicated time assigned to these prototyping tasks. The BI team should create a schedule with times and tasks for these business people to participate in prototyping.
- **Work incrementally**. Although there are software prototypes that are either throwaway or only work on part of an application, you're better off using a BI prototype that incrementally builds the entire BI application. This type of prototyping may be referred to as incremental, evolutionary, or extreme.
- **Time-box it**. The prototypes are rapid and time-boxed. This approach keeps both the developers and business people focused and better able to manage their time (the business people will have their "regular" jobs to complete besides the prototypes). Building the BI application in small and discrete increments enables a more thorough examination and validation when the prototyping is completed. Using this approach, you can develop and test incremental prototypes within hours.

The BI prototyping lifecycle as depicted in Figure 14.6 consists of the following tasks and feedback loops:

- **Define the prototype scope**. Determine the component that will be developed. Identify the participants (developers, reviewers, and manager). Define the objectives, deliverables sign-off criteria (success criteria), and the timeline. If the team is using a rapid prototyping or agile approach then this step may be completed in a single meeting.
- **Create BI application prototype**. Build the assigned component of the BI application. The assumption is that it will be built with the BI tool selected to deploy the BI application. The prerequisites are that a development environment exists and, at a minimum, sample data from the BI schema is available.

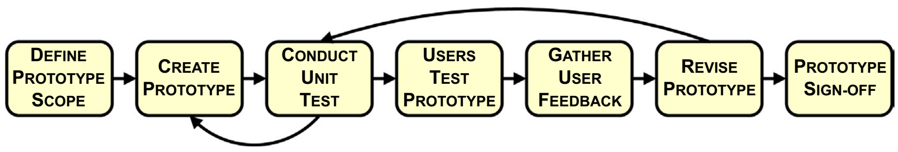


**FIGURE 14.6**

Business intelligence (BI) prototyping.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

- **Conduct unit tests**. The developer runs the prototype and examines test results. If changes are required, then revisions are made and the test is rerun. The prototype should only be passed along to the business people to conduct their own tests if it has passed the unit tests. If not, the developer informs the business people of any outstanding issues.
- **Users test prototype**. The business people test and assess the prototype. Over the course of the incremental prototypes, business people will test the data, UI, and types of analytics offered.
- **Gather user feedback**. The intent is to more actively involve business people in the development process to solicit their feedback and improve the overall BI application.
- **Revise as necessary**. If the prototype has serious flaws or shortcomings, then it should be revised and that prototyping lifecycle repeated. If the issues are minor or need to be investigated further, then implement any changes and test them in the final prototyping step that involves the overall BI application.
- **Prototype signoff**. For the individual components that are prototyped, a signoff indicates the next prototype can proceed. For the final prototype, the signoff indicates that the overall BI application has been validated and can proceed to the BI application testing lifecycle.

## BI APPLICATION DEVELOPMENT TASKS

BI prototyping is used for one or more of the following reasons:

1. Validate, expand, or gather business, data, and technical requirements
2. Determine feasibility of specific functionality
3. Examine effectiveness of components built
4. Engage business people to work with and provide feedback

The BI team may use prototyping to flush out the requirements (reason 1 listed previously) for the BI design. Prototyping may be used to determine either the BI content or the types of analytics that will be used in the BI application UI. If prototyping is used for design purposes, the BI team needs to document the design by using the BI tool's documentation capabilities. The prototype itself should be saved as part of the documentation.

BI prototyping is the best practice technique to develop BI applications (reasons 2-4 listed previously). The BI application will be divided into the following three components:

- Create data content
- Create data chart or visualizations for analysis
- Create overall BI application

The BI prototyping lifecycle is depicted in Figure 14.7.

### *Create Data Content*

The objective of this task is to get the data required for the analysis that will be performed using the BI application. At a minimum, this includes connecting to the required data sources, executing queries to gather the data, creating the schema used by the BI tool, examining the query results, and testing if the data is valid.

If the data sources are relational databases, then use open database connectivity (ODBC), java database connectivity (JDBC), or direct database connections. Nonrelational sources will require

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

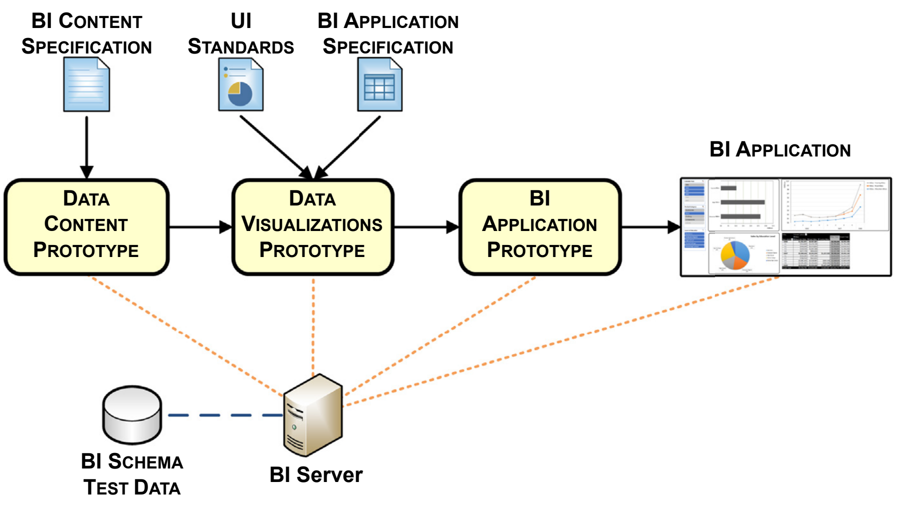


**FIGURE 14.7**

Business intelligence (BI) application development lifecycle.

connection information filled into source-specific application programming interfaces (APIs). BI tools typically provide all the prebuilt connection APIs that a BI developer would need, but there are ways to supplement those if necessary.

Typically, BI developers have several options for querying the data:

- Specify the data sources, often tables or files, and use the default setting of getting all the available columns or fields from that source.
- Specify the data sources and filter the columns (and any rows) that are retrieved.
- Write a custom-coded SQL statement (or whatever the equivalent is in the data source you are accessing) to get the rows and columns required.

When pulling data from more than one source table or file, use joins to gather the correct information. The typical options provided by the BI tool are as follows:

- The BI tool will join tables either by assuming that identical names in different tables imply a primary and foreign key relationship, or by querying the source database's metadata to identify columns designated as primary and foreign keys.
- The BI tool provides a visual application or a scripting language for the BI developer to designate primary and foreign keys. This may be the primary means to assign these keys or it may be how a BI developer would override assumptions that the BI tool had made regarding the key relationships.
- Write a custom-coded SQL statement (or whatever the equivalent is in the data source you are accessing) to join the appropriate tables.

The scope of creating and modifying the schema used by the BI tool, often called a repository, will depend on whether the business person accesses the schema to perform analysis or relies on prebuilt filters and drop-down lists to select data. Self-service BI applications such as data discovery, OLAP cubes, and pivot tables are the type of application requiring the business person to access the schema.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

When this happens, the BI developer may need to modify the BI tool schema to make it easier to use by doing the following:

- Creating user-friendly names for columns
- Removing columns that are never used
- Creating relationships not found in the data sources needed for analysis
- Creating new measures
- Implementing business algorithms
- Creating aggregations

This is typically performed using visual applications; however, some BI tools may require scripting.

### *Create Data Chart or Visualizations for Analysis*

To select the best data chart or visualization, the BI developer examines the type of analysis that the business person is going to perform and the data that it will use. Many BI applications, such as dashboards, will support multiple types of analysis and data sets, so more than one data chart or visualization will need to be selected.

After the data content has been loaded, the BI developer creates each data chart individually and tests how well it meets business requirements. After these tests, business people should get a chance to provide feedback on each of these data charts. This is not a formal user acceptance test (UAT), but rather one of several checkpoints in the development lifecycle to get feedback as the BI application is being incrementally built.

### *Create Overall BI Application*

Once all the data sources have been set up and each data chart has been built it is time to create the overall BI application. This means putting all the individual components into the overall BI application layout, establishing the navigation capabilities, implementing the UI standards and interconnecting the components as needed to support business requirements.

Once the overall BI application has gone through its prototyping tasks, then it will proceed into the BI application testing lifecycle discussed in the following section.

## BI APPLICATION TESTING

It is best to conduct BI application testing iteratively and incrementally as discussed in Chapter 18. BI testing is a collaborative effort between BI developers and business people whose involvement in working hands-on with the BI applications started with the BI prototyping tasks. As depicted in Figure 14.8, there are three testing phases: unit, BI applications, and BI systems.

The following tasks need to be performed:

- **Developer unit tests**. BI developers test each BI application after it has been built. The developers verify that the BI application operates in accordance with the BI specifications as they interpret them. This is a technical test of the application. If the developers discover any issues that would hinder business people from testing the application, then they should notify them and try to fix whatever is preventing the application from moving forward.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.

Sherman, Rick. Business Intelligence Guidebook : From Data Integration to Analytics, Elsevier Science & Technology, 2014. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/felician-ebooks/detail.action?docID=1832704.
Created from felician-ebooks on 2022-10-18 14:53:40.

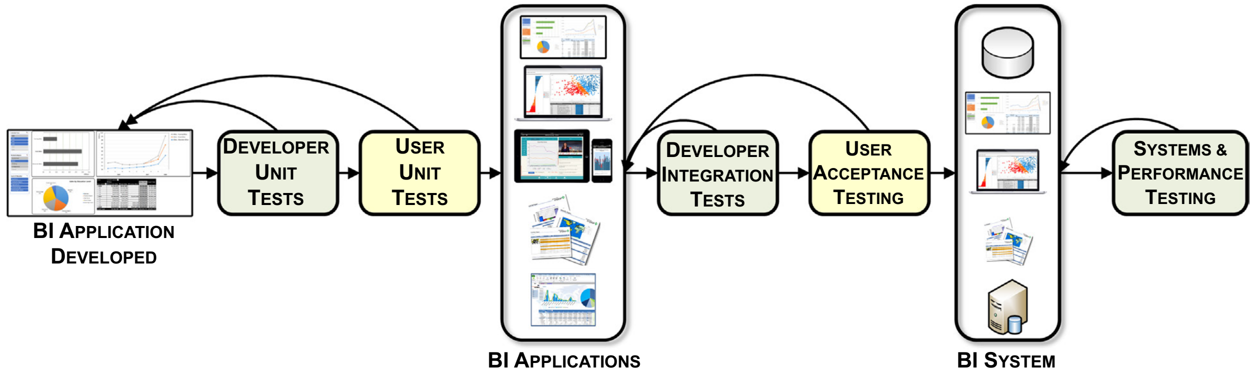


**FIGURE 14.8**

Business intelligence (BI) application testing.

- **User unit tests**. Business people test each BI application to verify that the application accesses the right data and performs the requested type of analysis. Business people will check that the UI's navigation and workflow support the work they plan to perform using this application. Also, business people should present any feedback, concerns, or issues they have with using the BI application. They and the developer should agree on whether the BI application is set to move on to the next phase or if fixes or modifications are necessary.
- **Developer integration tests**. BI developers will test all the BI applications that are planned to be released using "real" data rather than sample data that might have been used in unit tests. The real data should either be from a production environment such as a data mart, OLAP cube, or data warehouse, or from a testing environment that has the planned BI schema along with a representative subset of production data. The purpose is to verify that the BI applications can handle the real data and that the application functions correctly technically. If these conditions are met then UAT can commence.
- **UAT**. Using the same data as the developer integration tests, the business participants will validate that all BI applications access the right data, perform the appropriate analysis, and have UIs that are sound. The business people have the authority to provide the yes/no decision regarding the release of the BI applications. The business and technical sign-off criteria should have been established for the data, analytics, and UI before the UAT.
- **System and performance testing**. This involves testing the entire BI system including databases, data integration, and BI. The entire analytical lifecycle needs to be tested: sourcing, integrating, accessing, analyzing, and delivering the data. The system tests should involve several data-loading and business–analysis cycles. Performance tests should be conducted on the entire BI system including databases, data integration, and BI. If possible, create stress tests to simulate current estimated peak load and anticipated peak levels over the next 2 years.

A BI testing plan listing test cases with acceptance success criteria needs to be developed, documented, and signed off by the appropriate responsible parties prior initiating the BI test lifecycle.

Copyright © 2014. Elsevier Science & Technology. All rights reserved.