# UX Design Guidelines for Microsoft Dynamics CRM

**VERSION:** 1.0

**AUTHOR:** Roger Gilchrist, Hayley Bass, Prashanth GanapathyRaj
**COMPANY:** Microsoft Corporation

**RELEASED:** August 2015

**Microsoft Dynamics CRM**

## Copyright

# *Table of Contents*

# Introduction

Microsoft Dynamics CRM underwent a major user experience (UX) refresh with the release of Microsoft Dynamics CRM 2013. This was further enhanced with the release of Microsoft Dynamics CRM 2015. The key design principles behind the refresh were to keep the user experience simple, usable, modern, and fast. More details about these principles and the changes that were made can be found in this CRM 2013 UX Whitepaper.

The key objectives of this document are to:

- Illustrate how to use the new features introduced in or after Dynamics CRM 2013 to create a delightful experience to end users.
- Explain how designing solutions for versions in or after Dynamics CRM 2013 should differ from designing solutions for CRM 2011.

This document isn't intended to explain the technical details of the UX components. Instead, it's a collection of best practices based on real life customer scenarios.

# UX design process best practices

CRM implementation projects can follow any one of the many methodologies available. There are many variables that affect the impact and success of a CRM implementation. Irrespective of the methodology followed or the variables in play, there are some common traits across most successful CRM implementations. In this section, these general best practices for designing a user experience flow will be covered.

## Designing experiences for personas

One of the key traits of successful CRM implementations is the fleshing out of the personas using CRM, focusing on their goals and motivations. Based on this, one can identify the processes and insights that will be key to each persona. While going through this process, remember the following design decisions:

1) Design for the specialist role:

   Most implementations have specialist and generalist roles. Good UX design is seldom generic. It's advisable to design experiences optimized for each specialist role. Optimizing experiences for the generalist role makes it hard for the specialists to use the system, thus affecting the virtuous user adoption cycles. It's important to check for this pitfall right after the solution design phase.

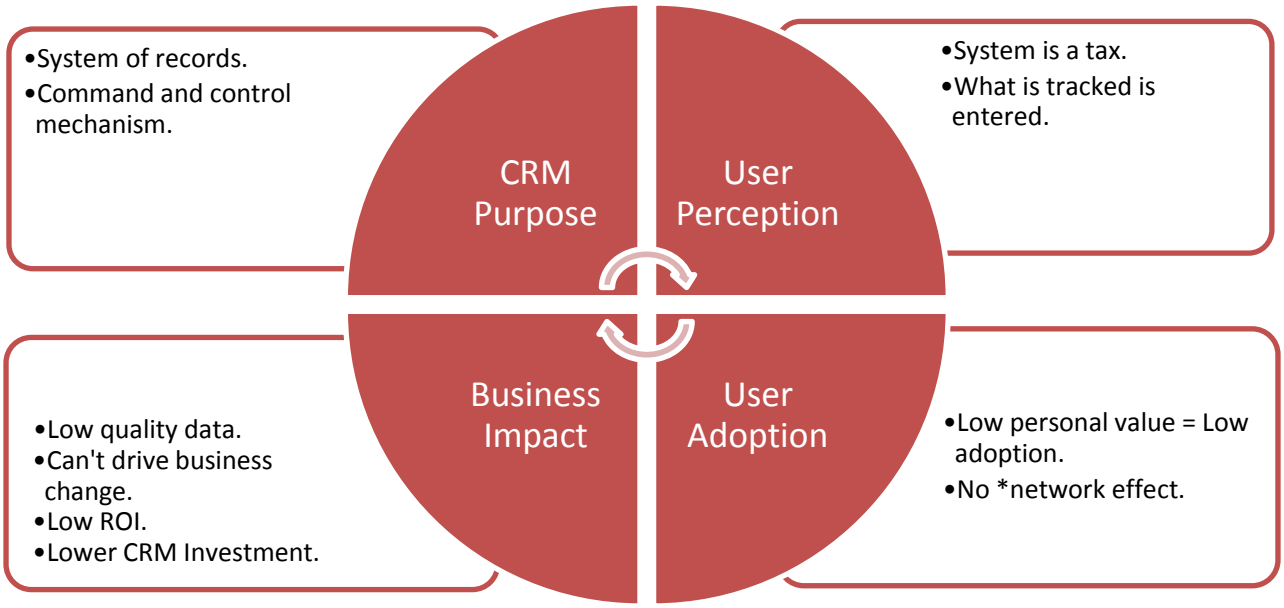
2) Design for the business user:

   Design of the user experience is targeted at the end business user following a business process. It is important to weigh each decision from the end user's point of view. A common challenge is the difference in how advanced developers or testers use the CRM system and how business users (the primary target user for the system) use the system. Business users and technical users have different goals and ways of working, which often lead to very different UX optimizations. The business user is the person the system should be optimized for. One way to mitigate risks is to make low cost prototypes and test them on real business users before finalizing designs. This can help discover usability issues early.

## Virtuous user adoption cycles

Another key trait of successful CRM implementations is the deliberate creation of a virtuous user adoption cycle. Adoption isn't a one-time goal at the beginning of go-live. Adoption is something that has to be monitored and tracked constantly so that changes can be made as needed.

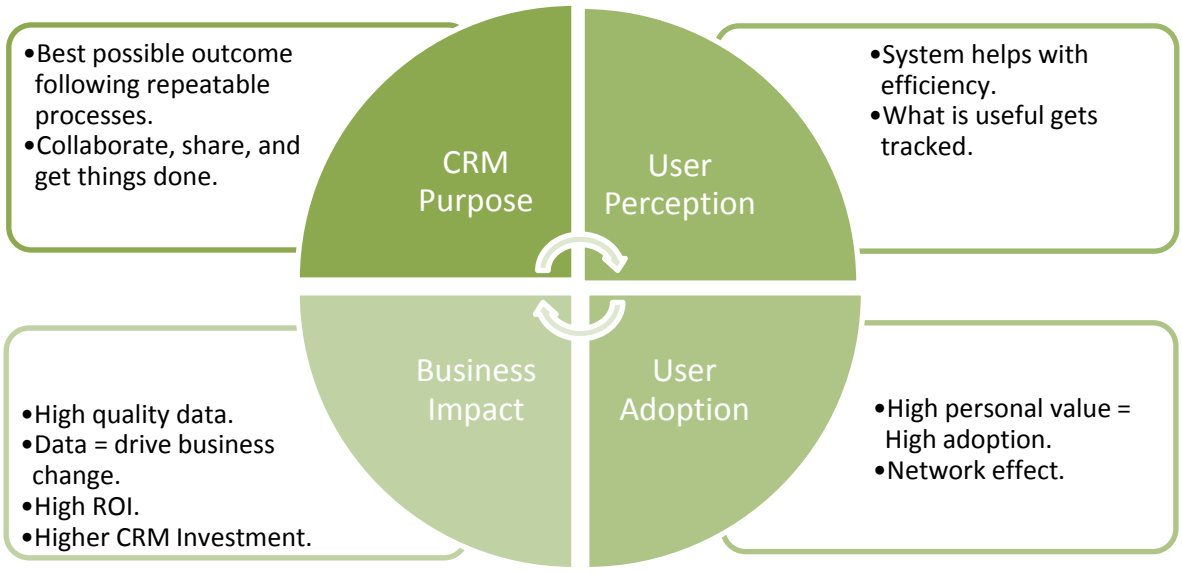
First, here's an example of a vicious cycle, where negative results reinforce each other to create an ideal climate for overall project failure.




* See Network effect for more information.

In the illustration, the CRM purpose is more aligned to management than it is aligned to end user needs. Such an objective leads to poor user perception, poor user adoption, and low business impact because users use it only at the minimum they need to in order to satisfy management. This, in turn, leads to lower investment in CRM.

Now, here's an example of a virtuous cycle, where positive results reinforce each other to create a positive spiral leading to sustained project success.




In this illustration, the CRM purpose is more aligned toward the end user's needs. Such an objective leads to better user perception as the users consider CRM as giving them value by making them more efficient. This leads to good user adoption and more quantity of higher quality data. This data can be used to drive business decisions. Businesses see a tangible impact and higher return on investment (ROI) and will invest more on a sustained basis in CRM systems.

The key take away here is this: It's very important to focus on the value CRM adds to the end users' day-to-day activities and how it helps them achieve their goals and objectives. If this shared purpose isn't established early and the focus isn't enforced through design and implementation decisions, poor adoption and overall project failure will likely follow.

## Focus on insights not data

In recent years, data has been seen as an ultimate solution to problems that businesses face. Terms like "big data," "data driven decision making," and such are used very often in business discussions. However, the truth of the matter is that a business might be sitting on a treasure trove of data. Unless they can figure out what the questions are that they want answered, the data is useless.

Successful CRM designs tend to focus more on the insights and on the actions based on insights, rather than showing raw data. Some examples of insights follow.

| What isn't appreciated | What goes wrong? | What would make things more efficient? | How to ease discovery? |
|---|---|---|---|
| •What do users not know about their customers or actions?<br>•What would change their interaction with customers? | •What is frequently done in error?<br>•What are the things that cause the worst impact, such as customer satisfaction or cost? | •Where does time or efficiency get lost?<br>•What behaviour can we drive to get to the right outcome first time? | •How to formalize organizational memory?<br>•How to discover what others are doing to avoid duplication? |

Let us look at an example of the question – "What do users not know about their customers or actions?"

Consider a retail bank named Contoso Bank and assume that financial fraud is the largest business concern. Now consider the situation when a customer calls the support desk about a wrong entry on their credit card statement. The wrong data to show to the support user here would be the number of support tickets raised by this customer in the last 30 days. Or the customer's last 10 transactions. If the key objective is to identify suspected financial fraud, the data to be shown here is all cases related to wrongful credit card charges in the last month from the customer's household. If this number is greater than a specified threshold, the system should issue an alert for suspicious activity. As you can see here, we are moving from static status data about the last five cases to data that is more typical to an insight that's actionable. Asking the right question helps to ensure that the system is designed to address the key organizational objective.

## Measure outcomes not actions

In large CRM implementations, the focus often is put on individual actions, rather than the outcome achieved. This can lead to very efficient individual actions. However, this isn't guaranteed to drive efficient outcomes. It's important to keep this in mind while you decide what to measure, which will determine if an implementation is successful.

Calculating the number of clicks to achieve a task is a measure of user interface (UI) efficiency, not a measure of user experience. This isn't to say that the number of clicks is not important. However, it is important to look at the big picture when making decisions on user experience.

For example:

- If one could increase First Time Close for service calls by 50% would it be worth adding two more clicks to the process?
- If you could increase your opportunity win rate by 10% by adding four extra clicks would it be worth adding them?

In both of these cases, if your main decision point had been based on a measure of number of clicks to achieve a task – an action – your answer would be no. However, if you're measuring the effect on business outcome, it would be a resounding yes. In particular, take the service call scenario. Each *initial* call would be two clicks more. However the *overall* number of clicks, when you take into account the follow up calls that would be needed to complete the customer request, would be lower. This broader view, not only of the complete scenario but also of the overall business benefit, is important.

Let us look at another example with Contoso Bank's retail operations:

- Individuals in the Contact Center are managing enquiries coming in via phone calls.
- Contact Center agents use CRM and CTI agent experience to provide a single customer view.

**Example:** International payments are process managed through CRM to multiple departments. Consistent information is captured and the process is visible.

**Previous process** – Paper form hand-off and basic system logging is used for processing. It took three to four calls with the customer and the internal department to complete requests and total time could take a few days. However, agents only clicked a few times to kick off each action with the next department.

**Current process:** All operations are managed in CRM with process flows. Most requests are handled and closed in one call as opposed to multiple calls. Initial call time is longer and adds about 20 more clicks, but the experience for the customer improved because the process is clear and easy to follow with fewer follow up calls needed. This has lowered the overall cost to the organization for customer request management.

# CRM UX components – usage recommendations

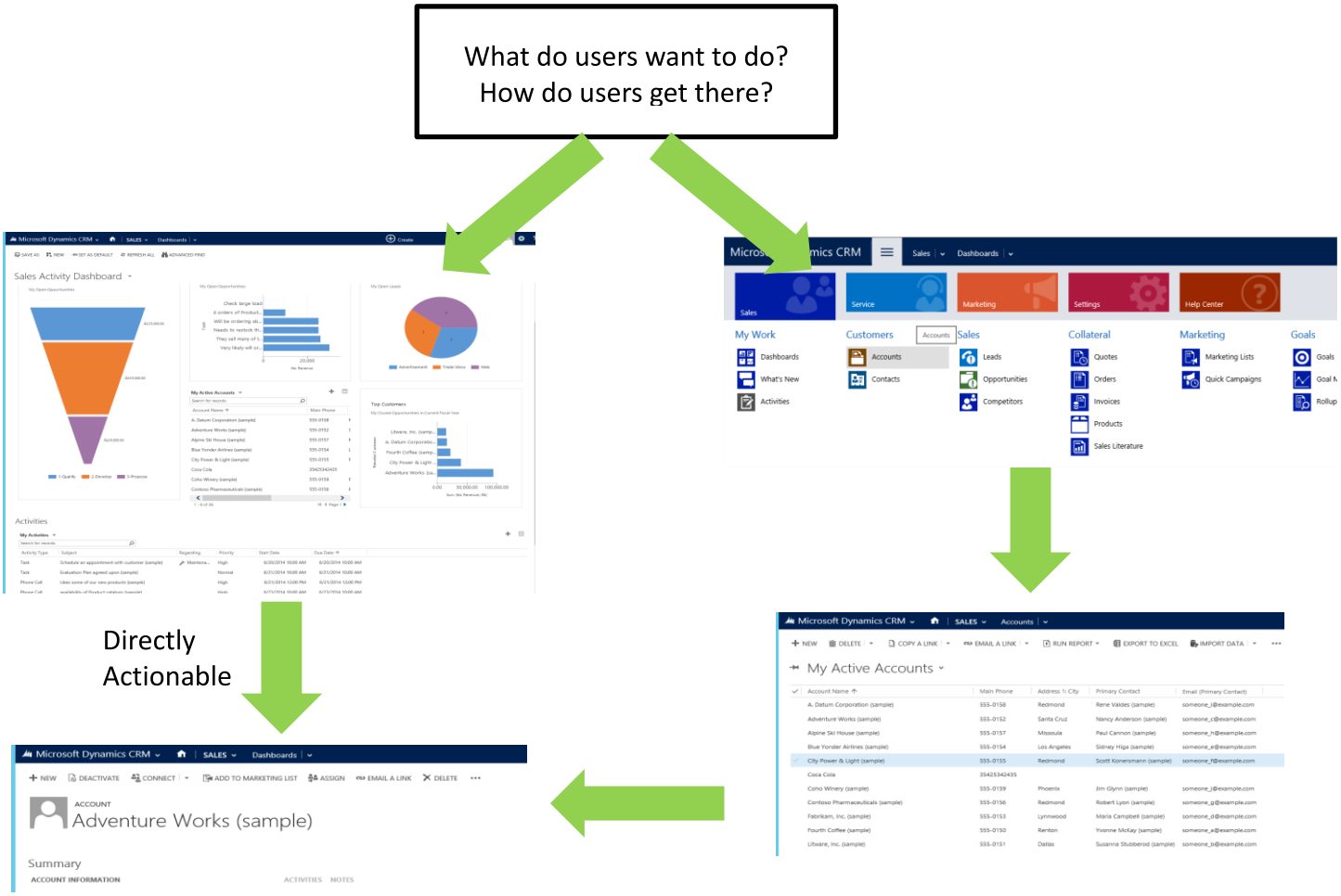
Dynamics CRM, as a platform, provides multiple options for the user to perform a task. It is the customizer's responsibility to ensure that the most efficient option is configured and recommended for usage. In this section, we recommend when to use which UX component in Dynamics CRM. This is based on the original intentions at time of design of these components, observed usage patterns on successful CRM implementations, and typical use case scenarios. There will always be edge cases that warrant a different approach from what is recommended. Hence it is important to understand the key principles and then apply those as appropriate to particular scenarios.

## Dashboards vs. navigation bar

Dashboards are a key component to the refreshed user experience. Role-based dashboards ideally should be designed for each role, answering key questions. These dashboards should be the home page from where 80% to 90% of day-to-day tasks for most roles should start. Ideally, all or most business flows should start from the dashboard. This was the key intended use for dashboards by design: to act as that initial gateway into a user's primary work. Consider specific roles individually and provide the information and insight vital to their role, rather than have a generic dashboard for all. The following table can help you decide what information to show on a role-based dashboard.

| Question | Visualization |
|---|---|
| What do I need to do to understand the state of business and my work? | Chart of important trends |
| What are the top tasks for the day? | Queue of activity I need to pick up, My Activities and tasks |
| Who do I care about? | View of key accounts and contacts |
| How am I doing? | Performance |

In contrast to dashboards, the navigation bar is used primarily to context switch. This is often misunderstood. The navigation bar isn't intended to be a part of a contextual business flow. It takes multiple clicks and visual searching to find the right area or entity to get to. These tasks make sense for data set exploration, for example, if the user is looking for a set of records via a view for an entity less commonly used. Note this is often an action a developer or customizer would want to do, highlighting the point earlier that the primary solution design should be optimized for the end user rather than necessarily for implementation tasks. However, this sort of task doesn't make sense to be part of a business process as it can cause friction and interruption in the user's interaction with the system. There are better ways to achieve the same effect when the data is needed in the context of their process.




## Subgrids vs. associated grids

Subgrids and lookups are used to provide a natural flow for quick in-context information discovery and consumption. Use subgrids on a form, when the related entity information in question is needed frequently and the number of records that need to be visible for effective consumption is low (<10). It's important to ensure that information that's key to the decision making process, like owner, is visible on the subgrids.

For example, if there are, on average, three to five opportunities per account and the key fields needed in the context of an account are only Opportunity Name, Owner, Estimate Revenue, and Estimated Close Date, it's ideal to show the opportunities associated with an account as a subgrid on the opportunity form. If it was necessary to

know what the status of all the opportunities were to determine if they're relevant to the current process, it would be important to add that to the subgrid as well. This avoids the frustration of users needing to open up each opportunity record individually to check the status; a simple change that has huge user experience benefits.

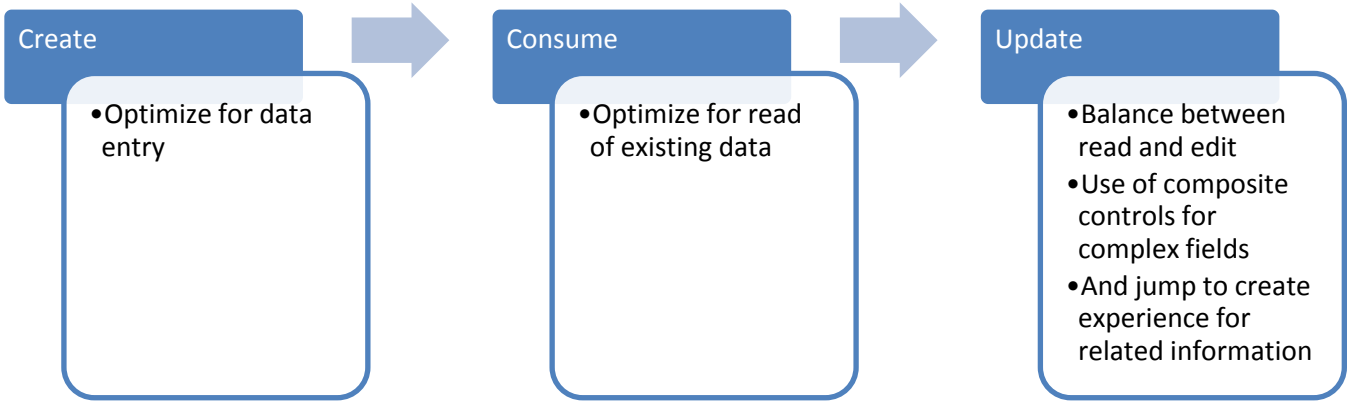
Associated grids are used to provide a more detailed immersive experience for related entity information. Use associated grids on a form when the related entity information in question is needed infrequently or if the number of records that need to be visible for effective consumption is moderate to high (>10). Associated grids take more effort and clicks to get to. However, they provide a dedicated experience to explore and consume a large number of records.

An ideal example is the out-of-the-box (OOB) associated grid for activities on the account form. A large account might have hundreds of activities regarding it. However, it is of interest only in a few situations like escalations or while analyzing the loss of a deal. Hence it makes perfect sense to show the activities associated with an account as an associated grid.

## Optimize for create and consumption experiences

Another key principle of the new user experience is recognizing that creating and consuming information are different models of working.

The new user experience allows each of these to be optimized for, recognizing that many adoption and usability challenges in the past were caused by a single form design intended to support both, but in reality optimized for neither.



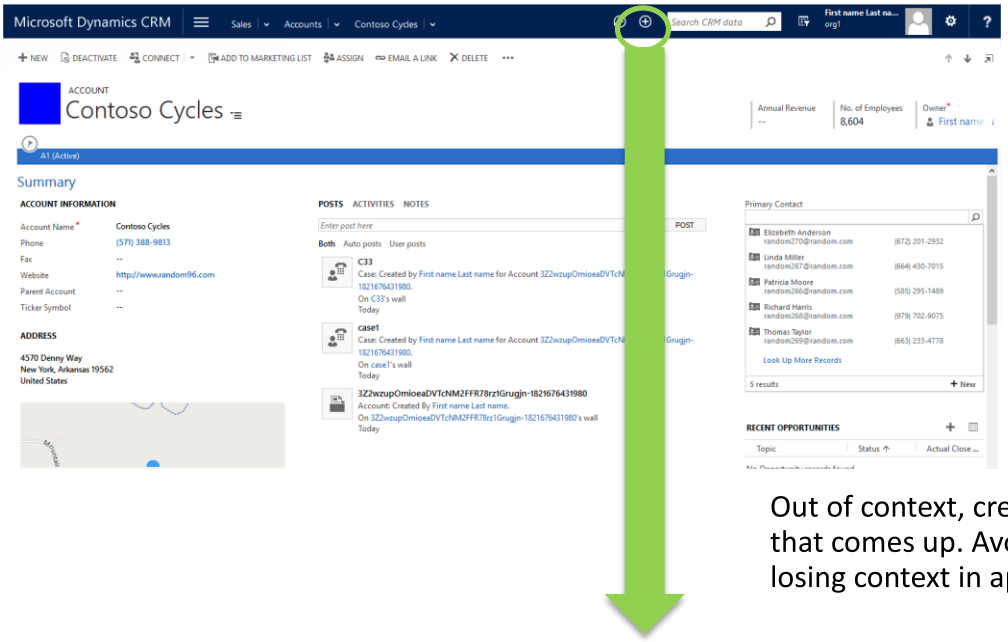

## Quick create form

The premise for designing the quick create form was the recognition of the fact that creating content is a different process than consuming content. The intent behind introducing the quick create form was to optimize the experience for creation. As the name indicates, this component was designed for creating content quickly. Hence it is important to only have minimum fields that would be always readily available at creation. The user can transition to the full form for further updates or edits if needed. Quick create forms also allow the main form to be designed and optimized for the consumption experience.

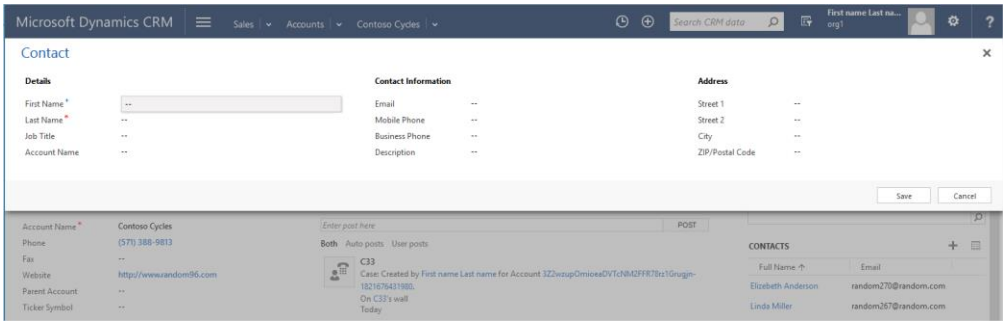
Quick create forms can be used in two scenarios:

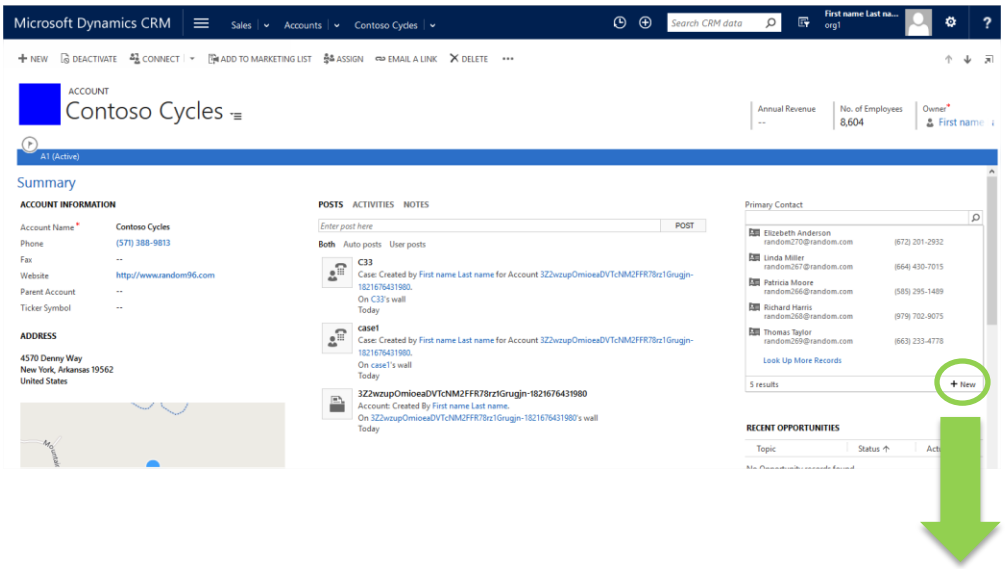
- To quickly create related information without losing flow. The form is launched from the subgrid on a form.

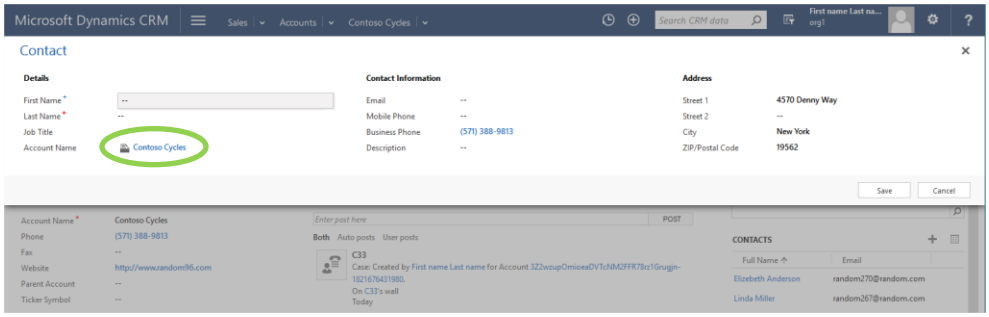
- To quickly create information in a new unrelated context. The form is launched from the Navigation Bar.




Out of context, creation of unrelated information that comes up. Avoid losing info, but also avoids losing context in application while doing it.





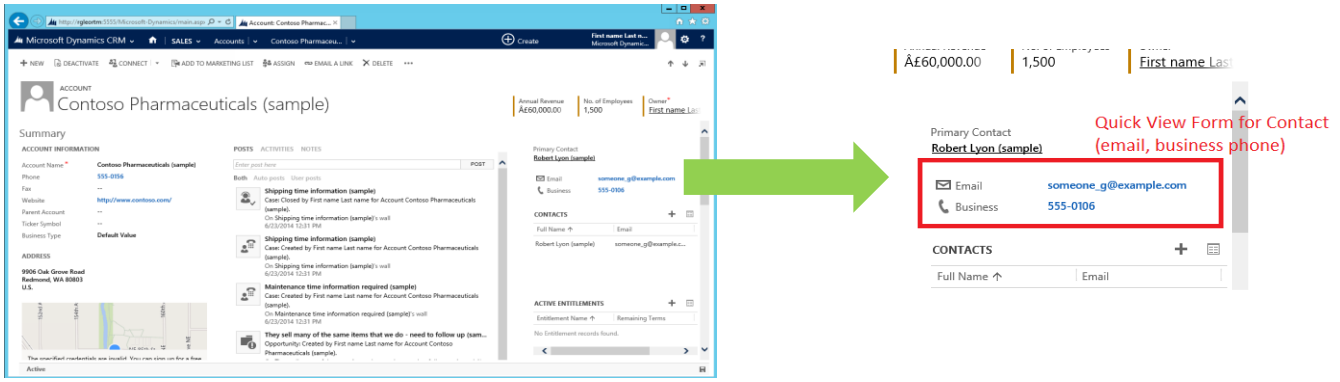

# Quick view form

Quick view forms were created to provide a quick preview of key attributes of a related entity without unnecessary navigation to the related record. This works by surfacing key information on a parent entity form through configuration. They weren't intended for building complete de-normalized forms where multiple fields from multiple related entities are all shown in the parent form. The preceding design option is possible, but it isn't recommended.

Quick view should be used sparingly:

- To show key information only, not replicate the entire record.
- To show information directly actionable or related to the outcome driven from the parent record.

For example, showing the email and phone number of a primary contact on an account form is an ideal use case for quick view forms where the likely action is to realize you want to contact the primary contact for the account. Rather than exploring the primary contact's full details on the contact form, quick views make it possible to directly contact them without that additional navigation. If you need to check out the contact's full details, that's still possible by navigating to the full contact form.




Note that it's possible to have a different quick view for the same entity with a different parent entity form. The more tailored the quick view forms are to the use case scenario, the more effective it will be. For example, the contact information needed in a quick view form in the context of an open invoice might be different from the information needed in the context of an account (as shown in the illustration). In an open invoice form, it might be important to show the credit limit or credit worthiness of the contact.

# Composite fields

Part of the focus on optimization of create and consume experiences was recognizing the different ways that structured data is edited as opposed to viewed.

Two particular types of data have been focused on in Dynamics CRM 2015 that highlight this principle:

- Names
- Addresses

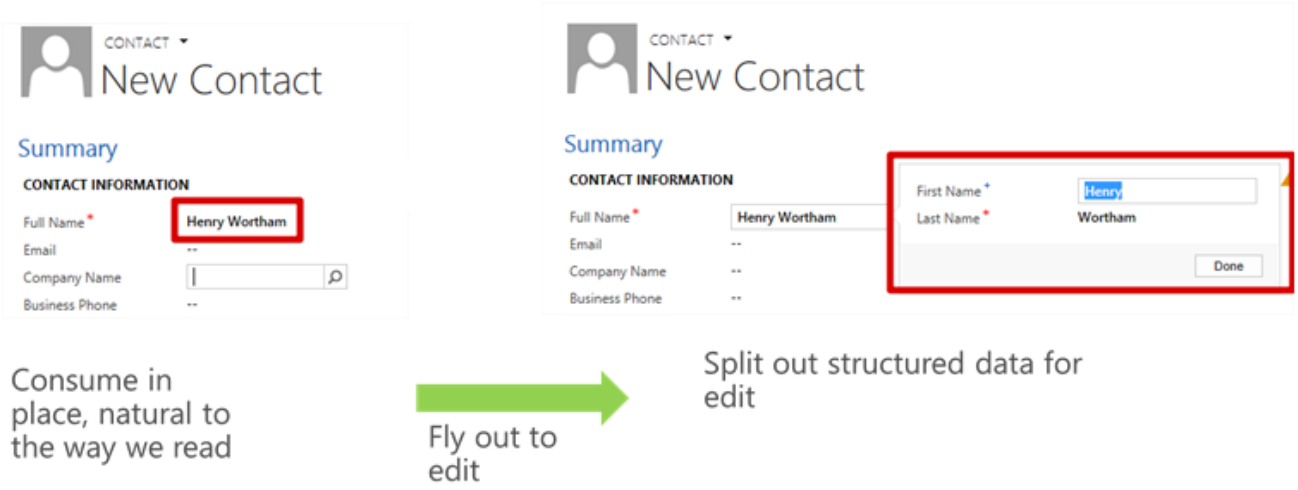
Take the example of names. We are used to consuming names in contiguous text, for example, such as the name Henry Wortham. Entering or editing name data however is typically necessary to do differently so that the correct structure is maintained such as being clear which is the first or last name, and whether there is a prefix or suffix.

Entering of a name is best performed therefore in structured form such as:

- First Name: Henry
- Last Name: Wortham

This avoids any ambiguity over which value reflects which structured value, but is not a particular easy form to read and consume.

To this end, on create forms the structured form of capture is always used for these field types. For full forms however, we noticed that in most cases the data would be consumed and, only occasionally, updated. Composite fields were introduced to assist with this hybrid experience, offering a more readable form most of the time, but offering through a “fly out” a structured edit experience when the data is to be updated.




With this approach, we’re able to provide:

- Optimized create experiences through structured capture on quick create forms.
- Optimized consumption experiences on forms when data is to be viewed through static forms of the composite fields.
- Optimized update experience on forms when data is to be edited by offering the fly-out composite field experience to offer structured editing.

# Process bar

The process bar was designed to provide an easy way for users to follow an established process. It's meant to encourage the right outcomes. It is not a wizard for data capture. It is recommended to use the process bar like a checklist to encourage the right practices and to enforce checkpoints at various stages. Therefore, it's recommended that the number of stages, and the steps per stage, are kept within numbers that are manageable and usable in the UI. This is especially important for mobile and tablet usage scenarios, where long or dense processes will more adversely affect usability.

Another key purpose of the process bar was to link together related entities that are tied to a process through automatic form transitions. This goes hand in hand with the inline navigation paradigm. It is most effective to design a user flow to leverage these UI transitions enabled by the process bar along with other natural navigation methods like lookups and quick views.

For example, the first step to update an existing case might be to associate a contact to the case and create one if needed via the quick create form. In this way, all components of navigation are user-focused for the user flow.

## Navigational flow

One of the key principles of the Dynamics CRM user experience is that of an intuitive flow through the application. The intention is that a user should be able to naturally follow a business process, with the system assisting them in achieving their goal, rather than the user feeling like they need to manipulate the software to achieve their ends.

Many of the enhancements have focused on this, in particular:

- In-place navigation rather than pop-up windows:
    - Previously, the use of pop-up windows meant that users lost context of their work as they opened and moved between windows.
    - In-place navigation allows the user to more intuitively flow forwards and backwards within a process without having to think about manipulating windows to get back to the information they require.
- Auto save:
    - This eliminates the need for the user to need to think or be prompted to explicitly save information as they progress through the application.
- Reduced scrolling and clicks
    - Changes to the overall presentation and flow of data reduces the need to scroll within long forms. This was a major cause of additional clicks to access information in previous versions of Dynamics CRM.

As will be shown in the next sections, a number of other features were added to assist with the presentation of data in context and the capture of information

# Design guidelines

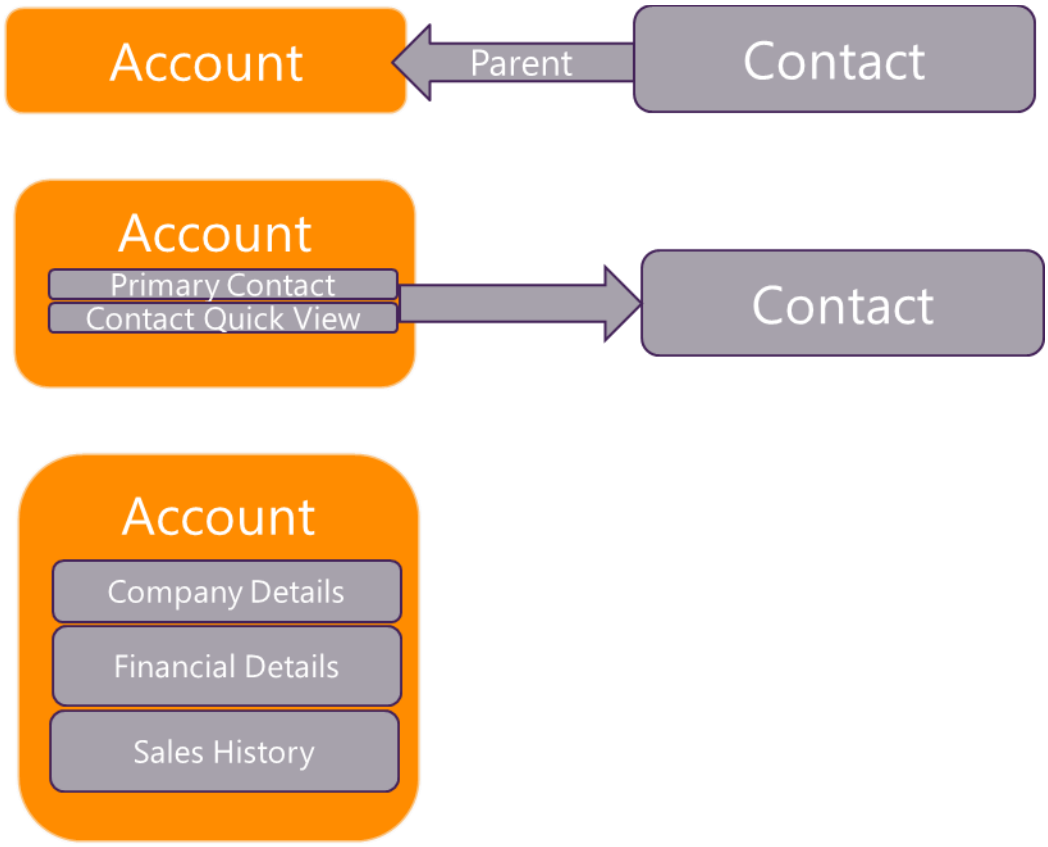
## Flow with data structure

When designing a system, it's useful to understand how people consume information. "Millers Law" is a valuable principle to understand and follow: Humans can process approximately 7 "chunks" of information at a time.

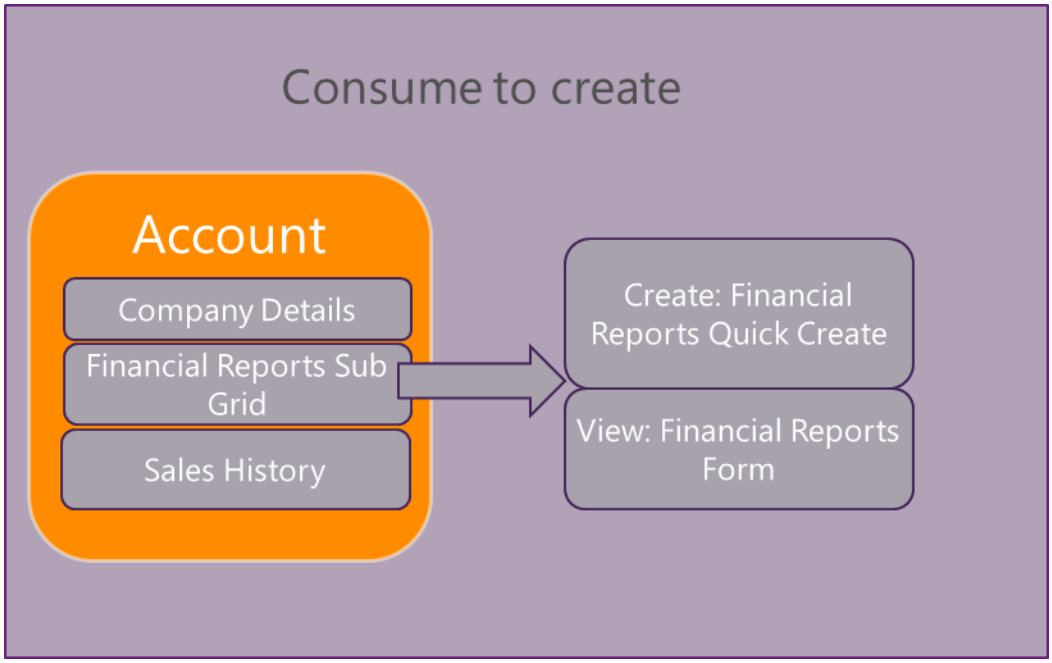
What this means in practice is that providing forms that contain long lists of fields is not an ideal way for users to consume the information. In reality, this forces them to subconsciously break the information into chunks of related and meaningful data so that they can consume it.

It is possible to assist with this and reduce the work of users to do this themselves, which both reduces the amount of effort needed, and therefore the frustration felt using the system, and also allows users to consume the data better.

This can be done by recognizing and representing natural structures and collections of related data together in the user interface. Breaking larger concepts, like an account, down to meaningful parts such as the company details, financial details, and contact information can assist with this. Each of these can then be represented either on a form or as separate entities.




Splitting data out into separate entities allows for a number of advantages. It's possible to separate the creation rather than consumption experiences for subsets of the data using quick create forms, such as in this example.




It's also possible to more easily control the visibility of data using standard product security mechanisms.

One of the reasons that data previously was often combined on a single form was to avoid the transition time required to load multiple forms and the navigational challenges of managing multiple pop-up windows. One of the enhancements of the new user experience is to make these transitions much more intuitive and simple. This means that the cost of the transitions is much lower. In fact, where a user is naturally having to switch from considering one concept such as account to another concept such as the primary contact, this transition can often be helpful to them in making that mental context switch.

The challenge to avoid is the mental jarring of the user during transition that often comes from having to mentally process how to make the switch or being delayed too long in making the switch. With the new user experience, both the time to make the switch and the more intuitive navigational flow avoids many of the issues that caused this mental jarring. This allows the system designer to take fuller advantage of the entity and navigational model to represent information. This is a much better model and experience for the user than trying to combine masses of data on a single form with the perception that it will improve the experience through less clicks or reduced navigation.

## Avoid "drop in" design

With the prevalence of a navigation-driven user interface design with previous releases, it's easy to fall back into the same approach, particularly when migrating a solution. It is important to recognize that the changes in the user experience are designed to drive a much more intuitive flow through the application, and a navigation-heavy approach isn't what many users prefer.

Recognizing and adjusting the design of an application to use navigation more naturally through the dashboards, forms, and views themselves and only using the navigation menu for context switches gives a much more natural experience for the user.

## Client-side scripting

It's fair to question the place for client-side scripting as a topic in a user experience document. However, it is here for a reason. The predominant cause for poor user adoption is poor performance. One of the major reasons for poor performance is poor scripting habits. Follow these guidelines to help ensure optimal user experience.

- Avoid scripting:

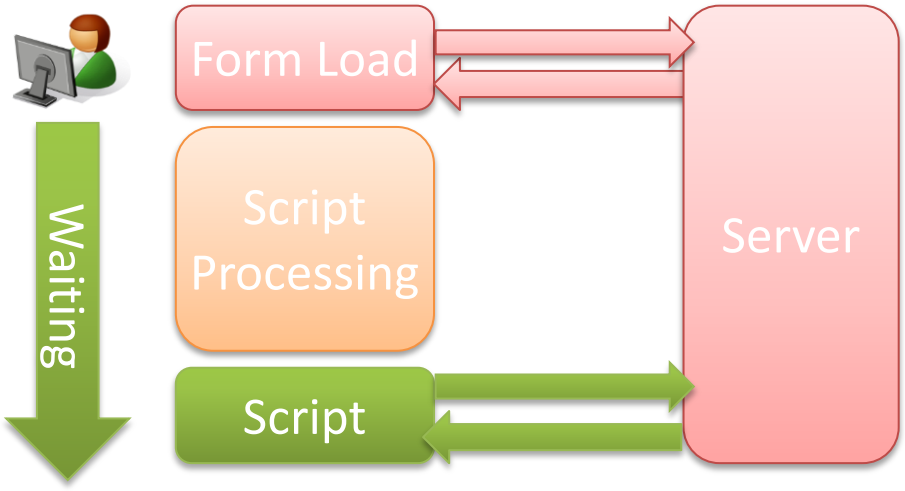
Try to avoid complex client-side scripting. Look at alternative options like server-side code or design change. For example, instead of showing and hiding some fields based on two roles, it might be better to create two roles and use two role-based forms.

- Onload event:

  Even if you choose to write client-side code, avoid functions with excessive wait times in the onload event. These delays directly add to the form load time. The higher the form load time, the lower the user's perceived application performance.

- Server callback functions:

  Server callback functions in client-side code cause the most delay. If a synchronous call is made and the client is waiting for a response from the server – to the user the client looks like it has stopped responding. Although the client is just waiting, the perception the user gets is that CRM is prone to hanging randomly.




## Custom theming

The primary purpose behind introducing theming is to help brand the CRM system along the same lines of other corporate line-of-business applications a customer might be running. This does have a tangible effect on user ownership and adoption of the system as well as enhancing the experience when the system is deployed in a customer-facing scenario.

Currently, theming allows for branding the application with a customer icon and changing accent colors for hover and selection of certain areas and entities. Theming isn't meant to enable changes to any of the current layouts or actions.

Dynamics CRM includes a default theme. Keep the following best practices in mind while you use the theming functionality:

- Accessibility:

  Be aware of the color contrast for new custom themes. Contrast ratio is an important measure of accessibility. Our OOB theme has the correct contrast ratios to ensure optimal usability. High contrast mode always uses the Dynamics CRM default color settings.

- Don't overuse colors:

Although every entity can be a different color, we recommend using one of two patterns:

- Make most entities a neutral color and highlight only the key ones.
- Make entities the same color when the entities have a similar purpose.

Also, keep the number of color groups low. If you use too many colors, the colors lose their significance and don't add to the user experience.

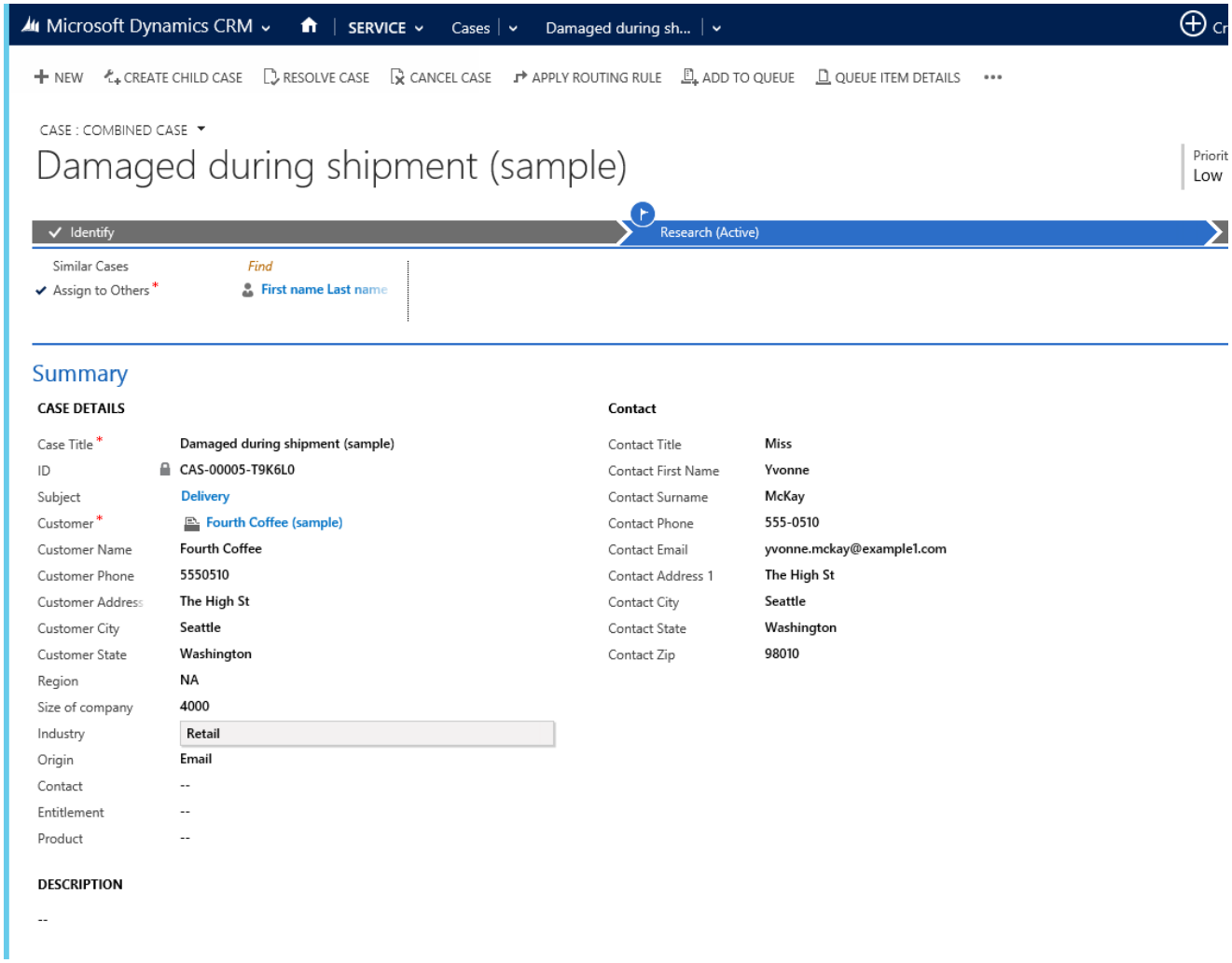
# Design example

As an example of the change in thinking, the following scenario highlights the difference in approach and the benefits that can be gained.

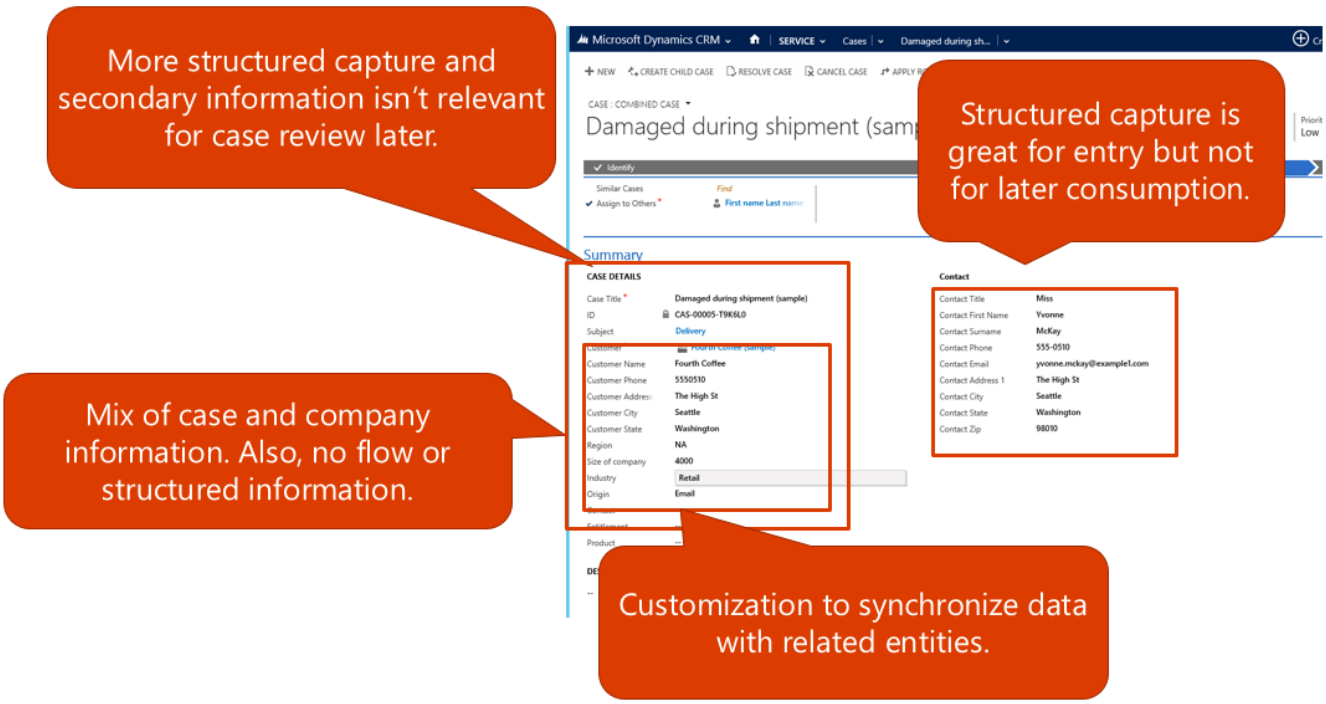
Consider a case capture scenario where:

- Most times the request was from a new account and contact.
- A single form for data input was required. However, this form would also be used later for action or review.

This approach led to this form.




In reality though, this design approach has a number of challenges.




Not optimizing the form for either creation or consumption leads to a suboptimal experience for both. Mixing data from the case, account, and contact entities all on the same form not only leads to a cluttered form, with a mix of information for the user to consume, but also requires significant customization work to achieve and keep the other entity data consistent.

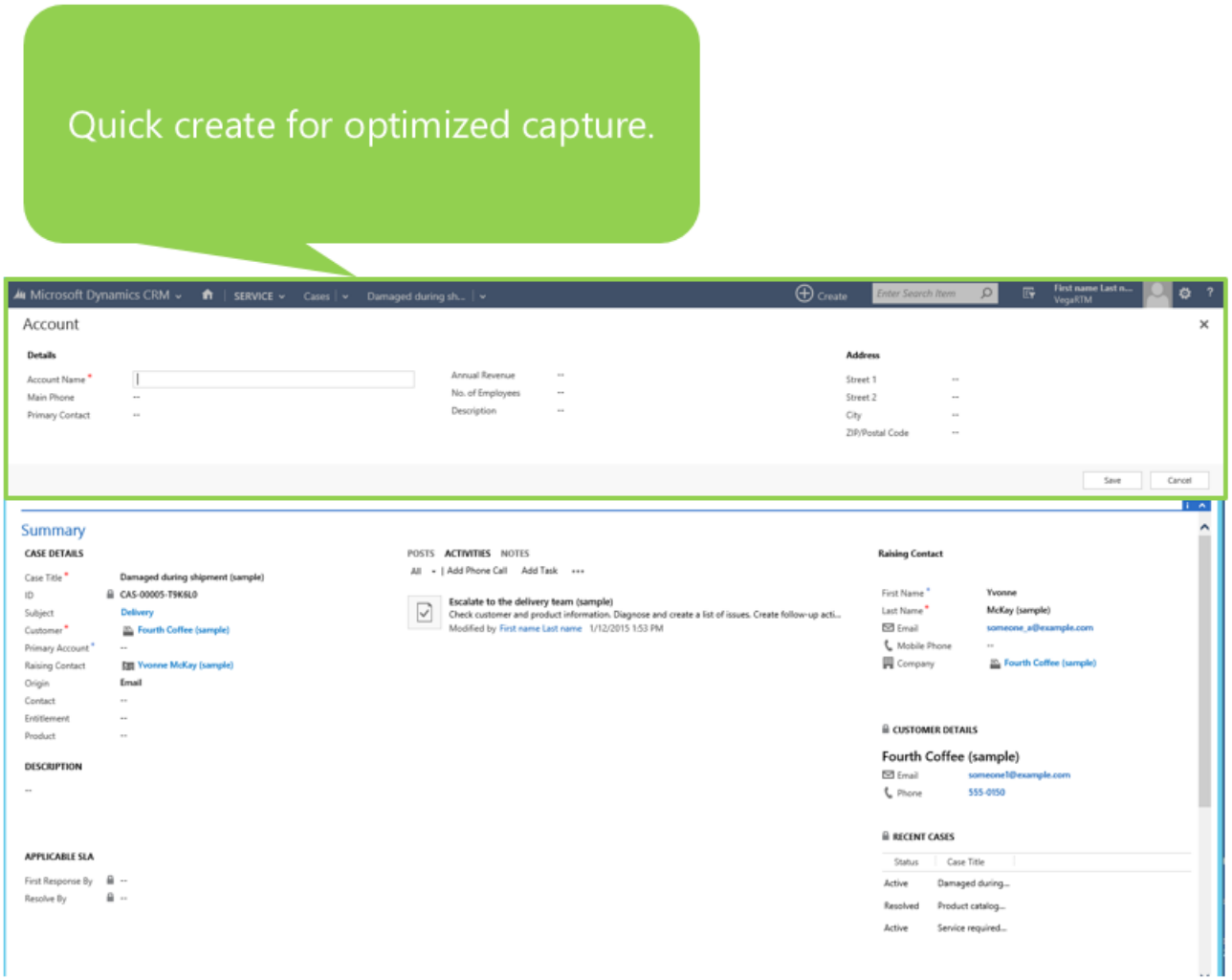
The primary requirement driving this was for single form capture. In reality, this is a design choice rather than a real user requirement. The correct requirement is for efficient information capture and processing, which can be achieved more effectively with the new user experience options.

The key design principles considered were:

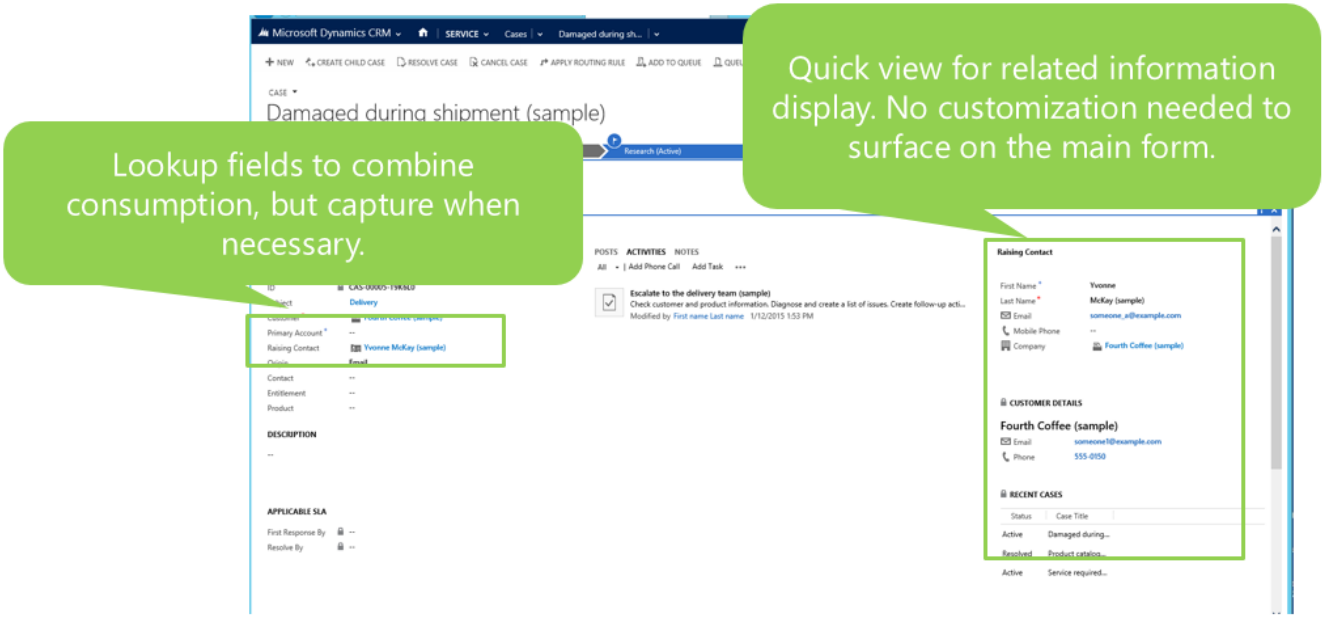
- Separate and optimize the creation and consumption experiences.
- Make capture of new information optimized for the new account and contact scenarios that are so common.
- Make the consumption of information for case review as efficient for the user as possible.

Using quick creates for related account and contact capture allowed us to optimize the creation experience and use structured information capture. This allows the user to do the following:

- Focus individually on capturing the information at hand, such as capturing information about the contact, and then switch context to capture information about the account.
- Capture record information that is used for back office processing, but won't typically be accessed during case review or handling, for example, the capture of company ID and tax status.




Using related entities, lookups, and quick views made the consumption experience as efficient as possible. Only showing key information on the quick views and transitioning to the full form for those entities for more detailed analysis aligned with the mental context switch, a user needs to fully consider the separate concept of an account or contact from the case handling they perform. This meant that the loss of a single form to gather and consume all the information was actually an advantage, focusing on the key information for case handling and allowing richer analysis on the occasions it was needed on the full account or contact form.




Another advantage of this approach is that it significantly reduces the customization required, enabling a configuration-only implementation. This reduces the overhead of maintaining the solution going forward. It ensures not only that upgrades are more straightforward, but also that as new capabilities such as applications for tablets and phones are introduced, those applications are more likely to simply work instead of require additional customizations or testing.

Aligning to the new product user experience gives not only a better experience for the end user but a more maintainable and flexible solution for the future.

# Summary

Before designing any CRM solution, think about user personas and objectives. Designing for the most specialized roles yields the best results. It's also important to focus on actionable insights rather than just on data. To measure the success of a CRM implementation, measure outcomes and not actions. Last by not the least – use the right CRM UX components in the right place of the user flow. Following the guidelines in this document can allow for optimal user experience and successful CRM implementations.