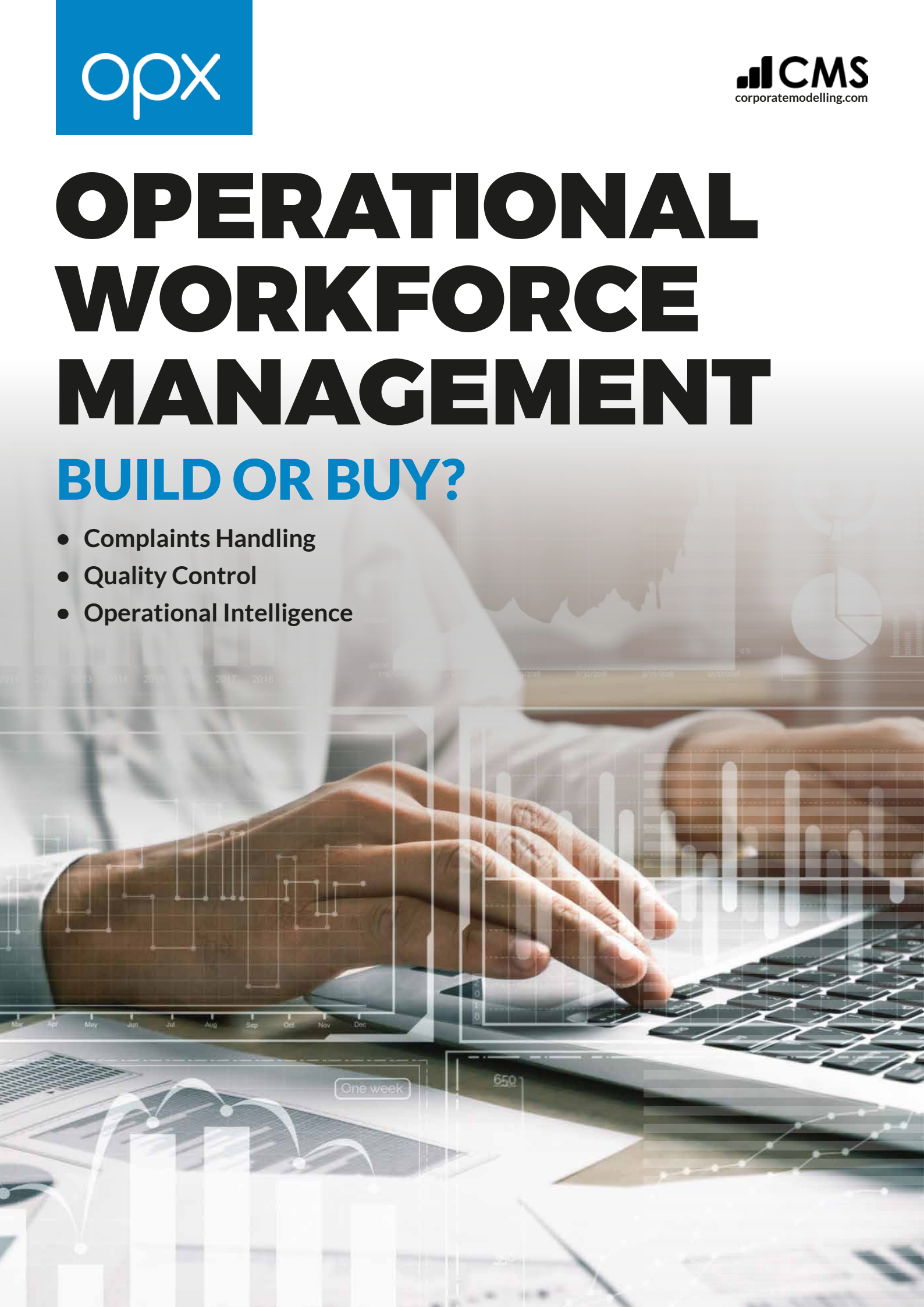
opx

CMS
corporatemodelling.com

OPERATIONAL WORKFORCE MANAGEMENT

BUILD OR BUY?

- Complaints Handling
- Quality Control
- Operational Intelligence



OVERVIEW

The world of operational management is changing with new initiatives coming on stream, customer demographics and the march towards self-service and digital platforms.

This paper looks at the pros and cons of creating your own internal systems to handle this and your existing business controls versus buying in all or parts of the system. It also looks at the challenges of both approaches.

Operational support systems need to provide a wide variety of features to cover the basics of providing operational control and operational management. A good starting point is measurement of the things you're trying to manage, as implied by Peter Drucker in his seminal work Management, Tasks, Responsibilities and Practices:

"If you can't measure it, you can't manage it."

What is it we need to measure? This is invariably:

- throughput
- quality
- costs

We refer to these as *the big three* because if the cost is right for the quality and throughput, the company must have processes, systems and people in harmony. It is not that they can't be improved, but that the external market pressures of customers and competitors do not need them to be improved.

Of course, it's rare for companies to be in balance, and the shareholders or managers happy with the returns. Even when they are, it is often going to hit new market or customer pressures or issues with growth and expansion.

So, what does this have to do with the *build or buy* decision? The *build or buy* decision is commonly a result of the company not being in perfect harmony of the big three, of wishing to expand, contract or break into new markets or product lines, and needing to change.

Rarely does a company say "*I'd like to change over a long, slow period*" – more often it needs to change quickly before the precipice looms or the expected precipice is upon them.

The factors we need to look at in a *build or buy* decision are:

1. **Time** – do we have the time available to do it ourselves?
2. **Knowledge and skills** – do we have the right people?
3. **Capacity** – do we have the enough capacity to run the build process as well as the day-to-day work?
4. **Risk** – can we manage the risks associated with a build project?
5. **Other factors** – can we define the requirements, costs, quality, support and continued development?

The rest of this paper works through these looking at both sides of the arguments. However, as you will see, instead of a *build or buy* statement, it should really be *buy and build* that most companies should adopt, mainly buying something and changing it if necessary – this is the COTS model, a *Customise Off The Self* approach.

FACTOR ONE: TIME FOR A PROJECT

When looking at the *build or buy* aspects of a new project for any of the major operational areas from Workforce Management through to Complaints and MIS, an important factor is often the time.

Many companies underestimate the time required to develop software. Where do all the time requirements come from?

Time is required for	Build	Buy
Define the business needs	High	Medium
Define the scope of the needs	Medium	Low
Define the requirements	High	Medium
Define project framework	High	Low
Define phases of delivery	Medium	Medium
Design the software	High	Low
Develop the software	High	Low
Test the software	High	Low
Business acceptance testing	Medium	High
Tailoring and configuration	Low	High
Maintaining the system	High	Low
Managing technical debt	Medium	Low

It can take far longer to decide what's needed when looking at business needs with a blank sheet of paper than when evaluating packaged solutions. Packaged solutions often come with a list of features and capabilities that can be enabled or disabled easily.

The scope of the build option is also less constrained: you can have requirements for anyone and everyone, and any area of the company, whereas packaged solutions often have a better definition of what they do and don't cover, and for whom.

As well as defining the requirements, we also need to define how the project will work. We need to define what phases are needed for delivery of the solution plus the installation and roll out. Unless you're the first user of a package, the supplier will normally have a framework and expected delivery and roll out.

When creating software, there is always design, build and test effort, but just like any other DIY task, even if you have the skills, the time taken to complete a rarely exercised skill is longer than the time taken by seasoned professionals. Most of us can plumb in a new washing machine but not at the speed of an experienced plumber.

Testing is a skill many companies lack. It takes time and effort, often involving automated test tools and scripts (more development and more testing not directly related to the business feature).

Often overlooked in a '*build your own*' option are the continued maintenance and technical debt, where technical debt is keeping up with changes to the underlying systems (even if simply security patches). Many underlying platforms have a shelf life of less than 2-3 years, so every 2-3 years you need to port to the operating system or database version with no other business benefit other than keeping the system running.

The winner of the time factor is BUY, as it's generally far faster to install and configure a package.

FACTOR TWO: KNOWLEDGE AND SKILLS

The second factor to consider is the knowledge and the skills to complete the mission.

Obviously, you have the business skills and knowledge to run your company or organisation, but can these be expressed clearly enough to the *build* team?

We often see the skills and knowledge needed as:

- business skills
- business knowledge
- market knowledge
- regulatory knowledge
- technical knowledge
- solution knowledge

A common scenario is that a few people in a business are superstars, others are competent, and many more may simply be capable. For the *build and buy* early phases, we need the superstars (for *build* more than *buy*) as they must impart their knowledge early, often starting with the simplest descriptions and levels of understanding to the in-house IT team.

External companies often occupy a space where they live and breathe their solution, and commonly pick up a breadth of knowledge in specific areas far in excess of all bar the superstars.

There is often also a hierarchy on the technical side of the equation such as:

- software architects
- software designers
- software engineers
- software developers
- software testers
- project and program managers
- technical authors & trainers

We often find that a business can support the cost of software developers, but rarely of architects and engineers, who command a far higher premium in the market. So much like the business profile, external software houses have software architects and software engineers. They often have program managers and project managers, and usually technical authors and trainers.

So, if the *build* team are going to create a solution to a similar level as a purchased one, they need to ensure they have the right skills or hire them in on contract. In terms of development, it's rare for market leading architects or software engineers to be working in a commercial or government organisation as the challenges, pay and rewards are often far higher for these rare skills in the software house and cloud B2B/B2C companies.

While it is possible to run a project without them, the quality, interoperability, maintenance and usability often suffer without these roles, as these roles tend to favour design and capability over function alone, making change easier in the long run.

The winner of the knowledge and skills factor is BUY, mainly because while the external provider does not know the business as well as your own superstars, they have a wealth of experience in the business practices in the market as well as overwhelming technical skills.

FACTOR THREE: CAPACITY

In a similar manner to the time factor, does the business have the capacity to deliver the project?

Let's assume a small operational module takes about 2500-man days for development from requirements to roll out and delivery. To deliver this project, the business needs to find between 10 and 11 heads depending on the absence, sickness and holiday statistics.

The company may have a full time IT organisation, in which case this may involve simply juggling the timing and priorities of projects in the IT department. This can affect the overall timing and delivery of this project or other projects, assuming the resources exist in the first place. If the business doesn't have the resources, they can be hired as contractors. However, it is generally unwise to have the whole project run by contractors.

An external software house is often on a maintenance contract and usually expanding their offering and hoping for future sales while contractors are often looking for the next engagement; unless you can persuade them to join as permanent employees, then the costs can spiral as the business tries to retain key resources, some of whom may be the only people with in-depth understanding of the new solution.

As mentioned before, systems are rarely – if ever – finished, so there is a need to maintain at least 10% of the FTE on the project permanently to keep up with fixing small issues and maintaining the day-to-day technical debt.

Software houses, however, tend to manage software development, deployment and maintenance as part of their core business. They will plan work that customers are often unaware of. For example, to ensure smooth transition for new versions of operating systems, databases and technologies.

Capacity for an external solution provider for your project is a fraction of your required capacity because:

- they have designed the system
- they have built the system
- they have tested the system

So, for your implementation they only need the capacity for:

- tailoring
- configuration
- installation
- training

Because payments are often related to deliveries in the areas above, external vendors tend to deliver the modifications in a timely manner. Unlike internal staff or contractors, they are driven to get a delivery out the door unless they are working on a time and materials basis.

So, because the purchased solution has had all the time-intensive steps of architecture, design, build and test built-in, the capacity for the external vendor is far lower than building one from scratch.

The winner of the capacity question is undoubtedly BUY, because otherwise you have a team in place doing little to nothing and awaiting work like this, while the external software houses use idle capacity to build new modules for sale.

FACTOR FOUR: RISKS

There are many risks in building new systems.

- Lack of *superstar* availability.
- Lack of business commitment.
- Never ending project, timelines slip, requirements creep, etc.
- Never starting development project, no time for full requirements gathering.
- Lack of skills to deliver the project, resource shortages.
- Lack of software tools.
- Lack of software licenses.
- Lack of hardware to deliver development and test environments as well as pre-production and live.
- Employee churn in the project (e.g. can we keep them for the timeframe?).
- Lack of regulatory knowledge.
- Key employee sickness or absence.
- Internal staff credibility with business team members.
- Poor architecture, causing interoperability and maintenance headaches.
- Poor design, causes maintenance and scalability issues.
- Lack of experience, slowing down project delivery.
- Markets or business move on before delivery of the project, requirements out of date.
- Requirements not fully defined.
- Process framework for internal sign-off results in waterfall delivery timelines.
- Development and delivery costs spiralling.
- Timelines missed.
- Quality poor.
- Technology choice (e.g. choosing access or end-user computing tools rather than real software tools such as SQL Server or Oracle due to cost and familiarity).
- Throwing resources to help with slippages (see *Mythical Man Month* by Fred Brooks).

It's easy to see that buying a commercial project can negate many of these as the software is ready to go (or ready to go with some tailoring or configuration) and the tens or hundreds of man years spent developing it are already taken into account.

The winner in the reduced risks factor is clearly BUY.

OTHER FACTORS: REQUIREMENTS, COSTS, QUALITY & CONTINUED DEVELOPMENT THROUGHPUT

The requirements are a key factor to the success of a project.

While the company may be able to write these down for the business aspects, they are often unable to create these for the technical side (other than the obvious, generic requirements such as scalability, robustness, security and low operating costs).

It is rare for internal teams to have in-depth skills in these areas simply because these skills command premium rates.

The same lack of practice will often be evident in:

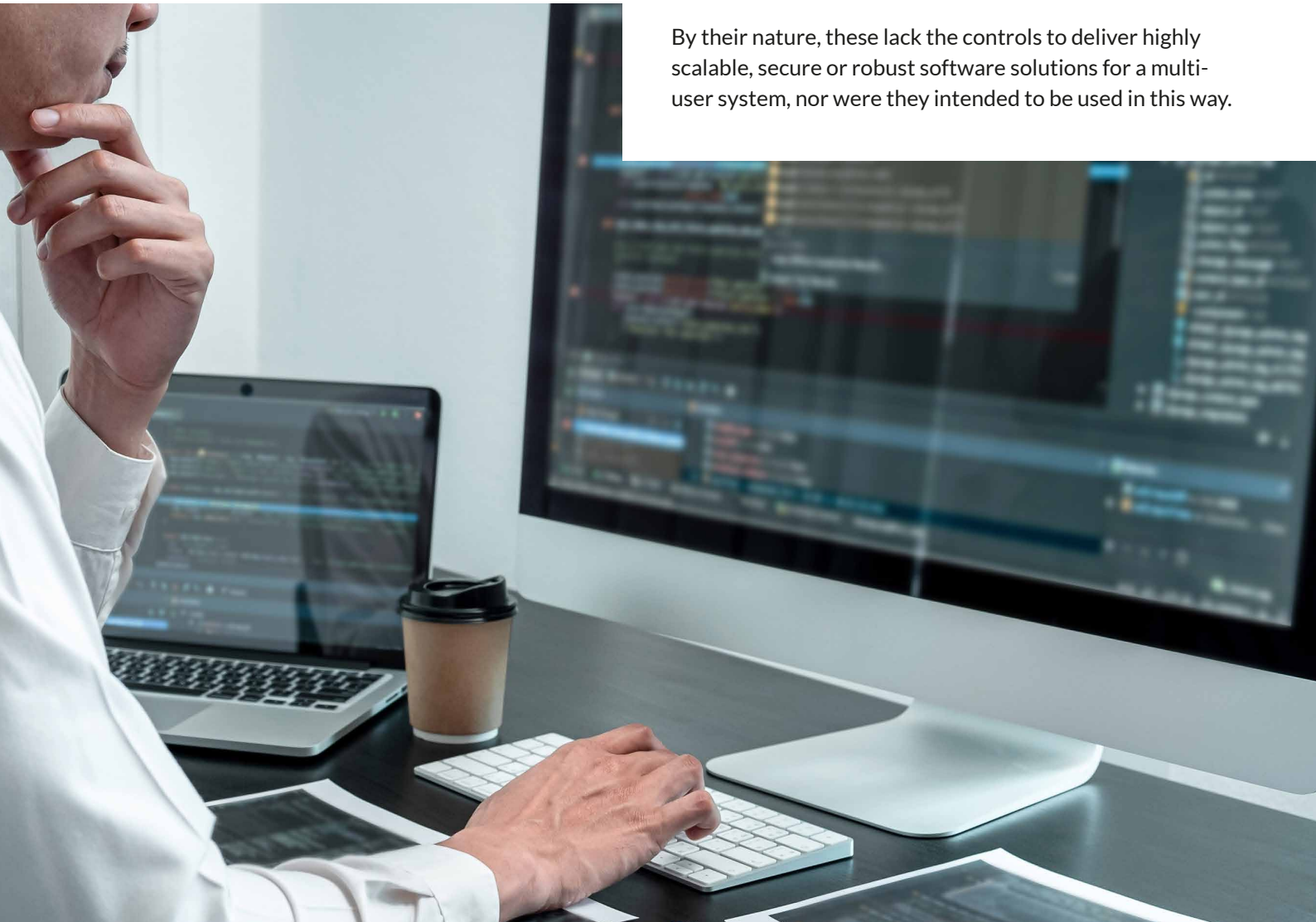
- software delivery cost management
- software development lifecycle management
- software quality checks
- software security, especially B2C or B2B delivery

It is often not understood that 70% of the software costs come after the initial implementation. Many of these are where the real impact of a lack of design and architecture can really hurt.

As pointed out by the FCA and others:

Robust, secure, scalable software is not developed in end-user computing environments.

By their nature, these lack the controls to deliver highly scalable, secure or robust software solutions for a multi-user system, nor were they intended to be used in this way.



THE BEST SOLUTION

The best solution is a mixture of do-it-yourself and off-the-shelf solutions known as COTS or *Customise Off The Shelf*.

With this scenario you benefit from:

- proper architecture
- proper design
- proper software tools
- ongoing maintenance
- ongoing external support where needed
- ongoing development of the core product

The business also benefits from being able to:

- define your own reports, dashboards and report models
- define your own tables to use with the tools
- enhance the solution with your own business knowledge and know-how

A key advantage is that many of the technology and security risks are already taken care of in the core solution.

SUMMARY

Unless the business has the internal resources and skills, it is rarely more cost effective to develop systems in-house, unless they are relatively simple.

Simple systems are often best used in cloud scenarios rather than developed in-house as external vendors commonly provide more features than one would develop if these had to be cost-justified internally.

More complex solutions are best developed externally and then tailored or customised to benefit from a better quality of design and architecture than one could justify creating with internal developers.

Want to know more about OPX?

Arrange a free demo and find out how OPX could support your business at: corporatemodelling.com/opx-demo

APPENDIX A - TOTAL COST OF OWNERSHIP CONSIDERATIONS

Acquisition and procurement

- Selection
- Upfront evaluation
- Purchase price
- Licenses
- Hardware
- Integration

Operation and Management

- Migration (data and users)
- Use
- Maintenance
- Upgrades
- Support services
- Training
- Software scaling
- Cost of customisation (change)
- Development, modification
- Carbon footprint

End of Life Management

- Retirement
- Disposal
- Migration (data and users)

Additional indirect costs may include:

- Costs incurred with another party to ensure the ability to meet Service Level Agreement targets for business-critical solutions)
- Unplanned costs, for example the possibility of unanticipated expenditure through compliance auditing and under-licensing.

ABOUT US

Founded in 2008, we have more than 35 years of experience in the field; we know the workforce optimisation space like the back of our hands.

The nucleus of the Corporate Modelling Services development team, based in Glasgow, UK and has been working together for over 15 years providing transformational software solutions to solve key business operations efficiency problems.

OPX is the result of over 200 man years of business focused enterprise software development and was conceived to provide a broad, functional, cost effective and yet easy to implement solution to aid the digital transformation of back office operations.

Every customer is unique. That's why we customise our OPX platform to fit every customer's needs precisely. Our Rapid Deployment Method (RDM) takes clients through the five steps of an OPX implementation in around 30 days.

OPX is proven to increase productivity and utilisation; reduce costs; improve cycle times and enhance customer experience.

MORE INFORMATION

For more information about OPX, please visit our website corporatemodelling.com

Corporate Modelling Services
Block 6, Kelvin Campus
Maryhill Road
Glasgow
G20 0SP
United Kingdom

CMS